HDV62A

High-Definition Video/Audio Capture Card

User's Manual

Manual Rev.: PRELIMINARY

Revision Date: July 3, 2012

Part No: 50-11246-PRE

Recycled Paper

Advance Technologies; Automate the World.

Revision History

Revision	Release Date	Description of Change(s)
PRELIMINARY	July 3, 2012	Pre-Initial Release


NOTE:

Please note that this is a PRELIMINARY version of the HDV62A User's Manual. While every effort has been made to ensure the contents hereof are currently accurate, subsequent releases may contain changes to the specification and operations, both minor and major, as well as entirely new chapters and modules not represented here.
For more information or if you have any questions, please visit our website at http//.www.adlinktech.com or contact your local Sales Center, as detailed in Getting Service.

Preface

Disclaimer

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.

Environmental Responsibility

ADLINK is committed to fulfill its social responsibility to global environmental preservation through compliance with the European Union's Restriction of Hazardous Substances (RoHS) directive and Waste Electrical and Electronic Equipment (WEEE) directive. Environmental protection is a top priority for ADLINK. We have enforced measures to ensure that our products, manufacturing processes, components, and raw materials have as little impact on the environment as possible. When products are at their end of life, our customers are encouraged to dispose of them in accordance with the product disposal and/or recovery programs prescribed by their nation or company.

Trademarks

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

Conventions

Take note of the following conventions used throughout this manual to make sure that users perform certain tasks and instructions properly.

Additional information, aids, and tips that help users perform tasks.

Information to prevent ***minor*** physical injury, component damage, data loss, and/or program corruption when trying to complete a task.

Information to prevent ***serious*** physical injury, component damage, data loss, and/or program corruption when trying to complete a specific task.

Table of Contents

This page intentionally left blank.

List of Figures

This page intentionally left blank.

List of Tables

This page intentionally left blank.

1 Introduction

1.1 Overview

The HDV62A full HD video/audio capture card, based on the PCI Express® x4 interface, enables acquisition of video and digital audio streams from both HD (high-definition) and SD (standard-definition) video at up to 170 MHz DVI input.

ADLINK's HDV62A not only delivers superior quality high-definition video data from DVI or HDMI, suitable for medical, scientific imaging and multi media device testing, but also provides an analog video decoder comprehensively supporting CVBS, S-video, analog RGB, and Component signal capture.

Equipped with a FPGA (field programmable gate array) and a 512 MB memory buffer, the HDV62A enables image streaming of specified target areas to a host PC, as well as real-time hardware color space conversion to offload repetitive tasks from the host CPU.

The HDV62A is equipped with the ViewCreator Pro® utility, enabling setup, configuration, testing, and system debugging without requiring any software programming. As well, ADLINK's WDM driver is compatible with Microsoft® Directshow to reduce engineering effort and accelerate time to market.

1.2 Features

- Uncompressed 1920 x 1080p, 60 fps video stream
- Support for CVBS, S-video, RGB and component video input
- Support for HDMI and DVI video input
- Support for HDMI, S/PDIF audio input
- PCI Express® x4 interface
- Hardware color spacezRAM frame buffer
- Configurable EDID
- Direct SDK support

1.3 Applications

The HDV62A is ideally suited to frame grab functions in a wide variety of applications, including:

- Medical imaging
- Scientific imaging
- HDMI video/audio testing
- Multimedia device testing

1.4 Specifications

1.4.1 Video

Format

Item	Specification
Digital	DVI 1.0, HDMI 1.3 (with HDCP)
Analog	R,G,B Y,Pb,Pr CVBS (NTSC / PAL) S-Video (NTSC / PAL)

Resolution (*with video detection)

Item	Specification
HDMI/DDVI (Digital)	YCrCb: VGA@60Hz(640 x 480) SVGA@60Hz (800 x 600) XGA@60Hz(1024 x 768) WXGA+ @60Hz(1280 x 768) WXGA @60Hz(1360x768) WXGA @60Hz(1280 x 800) SXGA @60Hz(1280 x 1024) UXGA @60Hz(1600 x 1200) WSXGA+@60Hz(1680 x 1050) 525i@30(720 x 480) 625i@25 Hz(720 x 576) 720p@50/60Hz(1280 x 720) 1200P@50/60Hz(1920 x 1200) 1080i@25/30Hz(1920 x 1080) 1080p@24/25/30/48/50/60Hz(1920 x 1080)
RGB (Analog) (via extension bracket)	VGA@60/@75/@85Hz(640x480) SVGA@60/@75/@85Hz(800x600) XGA@60/@75/@85Hz(1024x768) WXGA+@60/@75Hz(1280x768) WXGA@60/@75Hz(1280x800) SXGA@60/@75Hz(1280x1024) UXGA@60Hz(1600x1200) WSXGA+@60Hz(1680x1050)

Item	Specification
YPbPr (Analog) (via extension bracket)	525i@30, 525p@60 625i@25, 625p@50 720p@50/60 1080i@25/30, 1080p@50/60
CVBS & S-video (Analog) (via extension bracket)	SDTV:720 x 576 720 x 480

General Video

Item	Specification
Pixel Output Format	RGB:32bit RGB* 30bit RGB* 24bit RGB YCbCr:24 bit 4:4:4 YCbCr 16 bit 4:2:2 YCbCr Monochrome:8-bit Y
Area of Interest	One AOI range, range (W:16x, H: 2x)
Capture Modes	▶ Free Run ▶ Snap: software
Video In	▶ DVI-I Dual Link connector: DVI, HDMI (Only single-link signal) ▶ SMB,BNC connector (External bracket): Y,Pb,Pr ▶ 10 Pin Box header, 9-pin D-Sub Female connector: (I/O bracket): CVBS,S-Video

Some computers may experience minor (10% or less) reduction of fps performance when 32bit or 30bit RGB output settings are in effect.

1.4.2 Audio

Item	Specification
Digital Format S/PDIF (IEC60958-compatible)	Sampling Rate: 16, 22, 32, 44.1, 64, 88.2, 96 KHz Input Bit format:16 bit, 20 bit, 24 bit Output Bit format:24 bits Channels:2.0 (2) Note: no A/V sync
Digital Format HDMI (HDMI1.3)	Sampling Rate: 32, 44.1, 48, 88.2, 96, 176.4, 192KHz. Input Bit Format:16 bit, 20 bit, 24 bit Output Bit Format:24 bits Channels:2.0 (2), 5.1 (6), 7.1 (8) Note: no A/V sync
Analog Format	N/A
Capture Modes	▶ Free Run ▶ Snap: software
Audio in (SPDIF)	▶ Toslink connector (for optical fiber cable) ▶ RCA connector (for 75-ohm coaxial cable)
EDID	▶ EDID 1.3 (Audio and Video) ▶ Software configuration

1.4.3 General

Item	Specification
RAM (Frame Buffer)	DDR2 667 512MB
Flash Memory	▶ Dual (Golden flash and working flash selectable by switch) ▶ Software download
Power Consumption	▶ Max. @ +12V ▶ Max. @ +3.3V
Board Dimensions	174.62 x 111.15 mm
Board	▶ Half-length PCI Express 1.0a x4 compliant ▶ ROHS Compliant

Item	Specification
Operating Temp.	0 to 55° C
Software Support	XP / Win 7, 32/64 bit, DirectShow

1.5 Schematics & Connections

NOTE:

Please note that all dimensions shown are in mm.

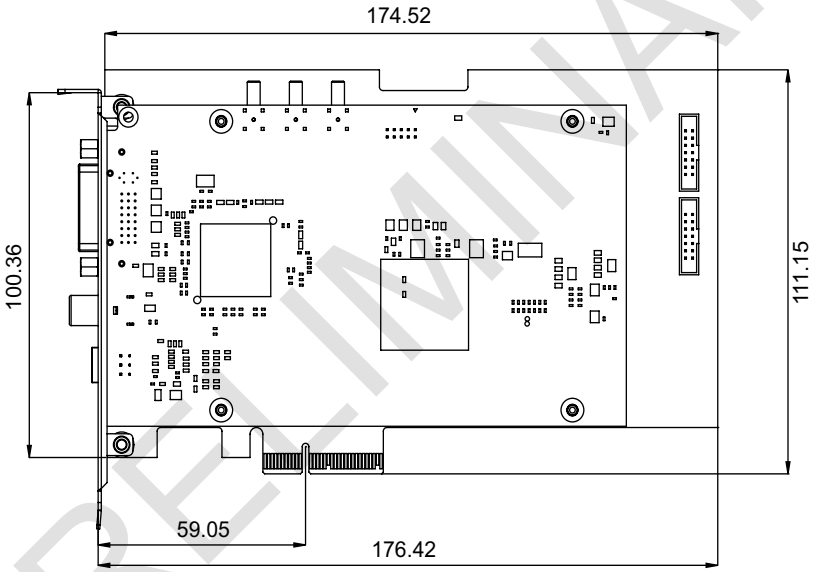


Figure 1-1: HDV62A PCB Overview

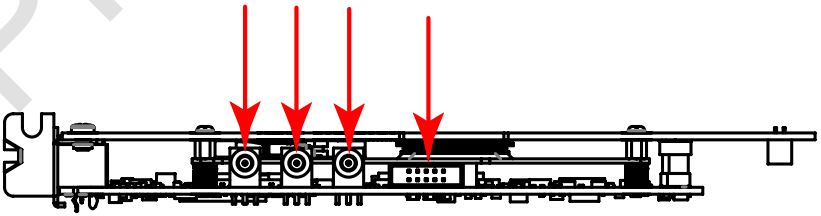


Figure 1-2: HDV62A PCB Side View (showing I/O bracket connectors)

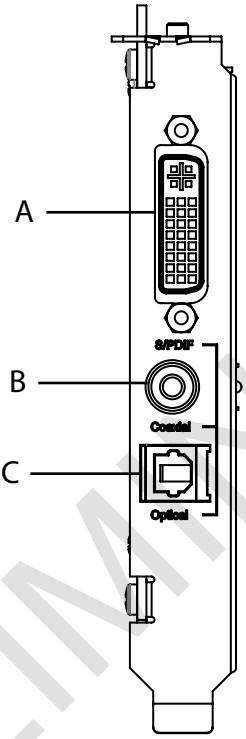


Figure 1-3: HDV62A I/O Panel

A	DVI-I for DVI, HDMI and RGB
B	RCA for S/PDIF
C	Toslink for S/PDIF

Table 1-1: HDV62A I/O Legend

1.5.1 DVI-I

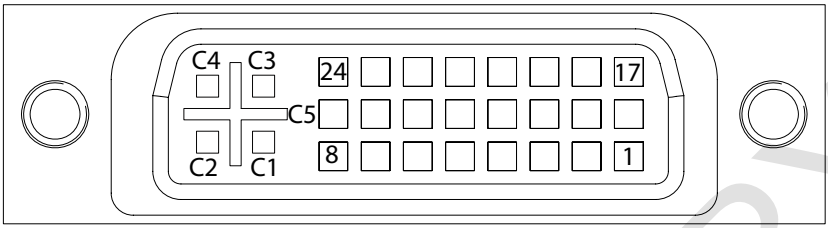


Figure 1-4: DVI-I Connector

Pin	Signal	In/Out	Pin	Signal	In/Out
1	T.M.D.S data2-	Input	16	Hot plug detect	Output
2	T.M.D.S data2+	Input	17	T.M.D.S data0-	Input
3	T.M.D.S data2 shield		18	T.M.D.S data0+	Input
4	NC		19	T.M.D.S data2 shield	
5	NC		20	NC	
6	DDC clock	Input	21	NC	
7	DDC data	Input	22	T.M.D.S clock shield	
8	Analog Vertical Sync.	Input	23	T.M.D.S clock+	Input
9	T.M.D.S data1-	Input	24	T.M.D.S clock-	Input
10	T.M.D.S data1+	Input			
11	T.M.D.S data1 shield		C1	Analog Red	Input
12	NC		C2	Analog Green	Input
13	NC		C3	Analog Blue	Input
14	+5V Power	Output	C4	Analog Horizontal Snyc.	Input
15	Ground (Return for +5V Hsync and Vsync)		C5	Analog Ground (Return for Analog R, G, and B)	

Table 1-2: DVI-I Pin Connections

1.5.2 Toslink Optical and RCA Connectors

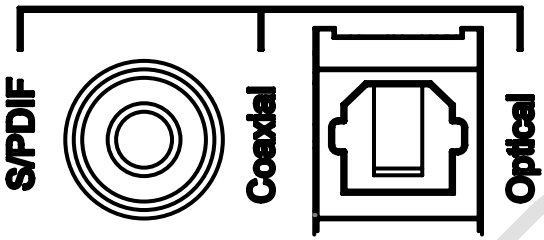


Figure 1-5: Optical and RCA Connectors

Pin	Signal	In/Out
RCA	S/PDIF	Input
Toslink	S/PDIF	Input

Table 1-3: Optical and RCA Pin Assignment

1.6 Switch Settings

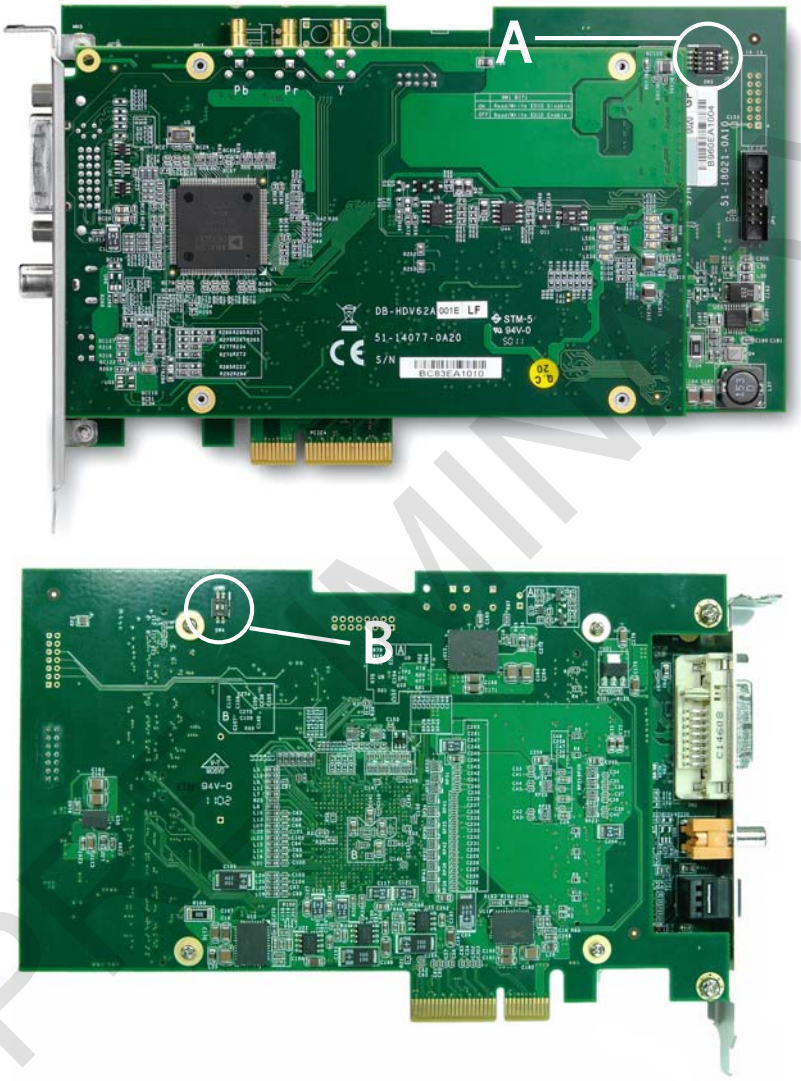


Figure 1-6: Switch Locations on PCB

A	Card ID switch
B	Flash selection switch

Table 1-4: Switch Locations Legend

1.6.1 Card ID Switch (SW3)

NOTE:

Please note: all settings shown 4321, and ON=0, OFF=1

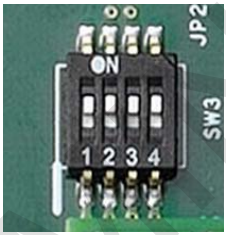


Figure 1-7: Card ID Switch

Card ID	Setting	Card ID	Setting
0	0000	8	1000
1	0001	9	1001
2	0010	10	1010
3	0011	11	1011
4	0100	12	1100
5	0101	13	1101
6	0110	14	1110
7	0111	15	1111

Table 1-5: Card ID Settings

1.6.2 Flash Selection Switch (SW4)

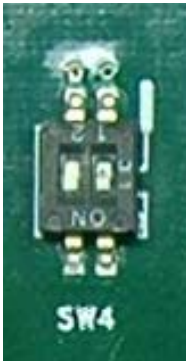


Figure 1-8: Flash Selection Switch

Flash	Setting
Boot Flash	Switch(Bit2,Bit1)
Working Flash	OFF,OFF
Golden Flash	ON,ON

Table 1-6: Flash Selection Settings

1.7 Optional Connections

ADLINK provides expanded connectivity with a variety of optional cable solutions, available to HDV62A users.

1.7.1 YPbPr/S-Video/CVBS I/O Bracket

3x SMB to BNC + 1x 10-pin to CVBS/S-Video cables with I/O bracket allow external connection of YPbPr and CVBS/S-Video signal sources (See “HDV62A PCB Side View (showing I/O bracket connectors)” on page 6.for connector location on PCB).

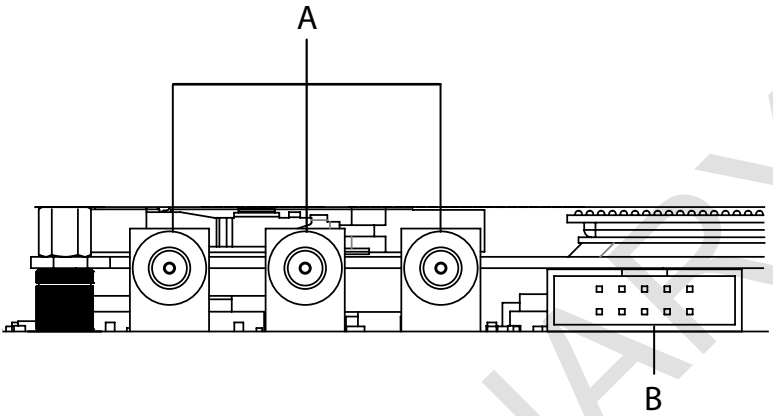


Figure 1-9: I/O Bracket PCB Connections

A	SMB
B	10-pin

Table 1-7: I/O Bracket PCB Connection Legend

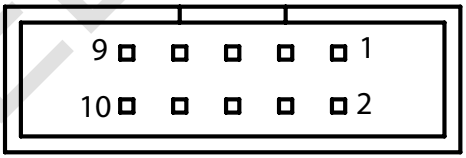


Figure 1-10: 10-pin Box Header on PCB

Pin	Signal	Pin	Signal
1	Y(S-Video)	2	CVBS
3	C(S-Video)	4	Video Ground

Pin	Signal	Pin	Signal
5	Video Ground	6	Video Ground
7	NC	8	NC
9	NC	10	NC

Table 1-8: 10-pin Box Header Pin Connections

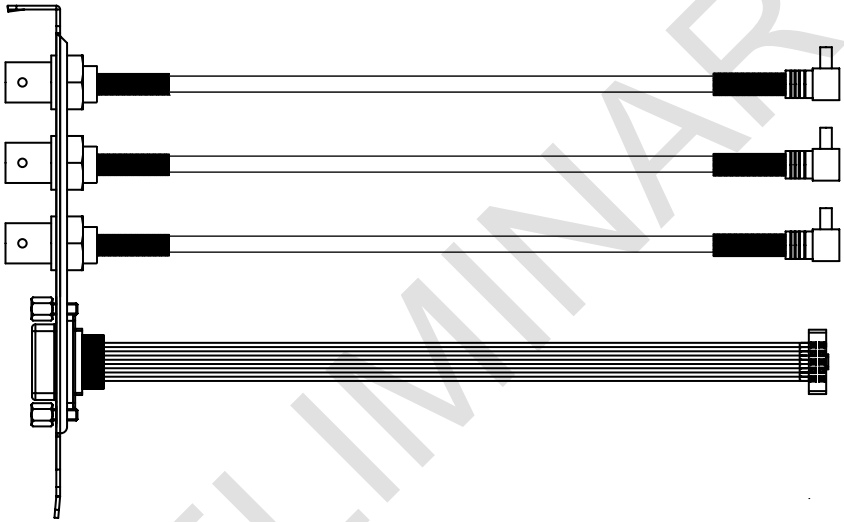


Figure 1-11: I/O Bracket

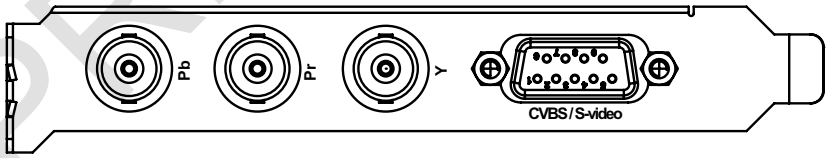


Figure 1-12: I/O Bracket External Connections

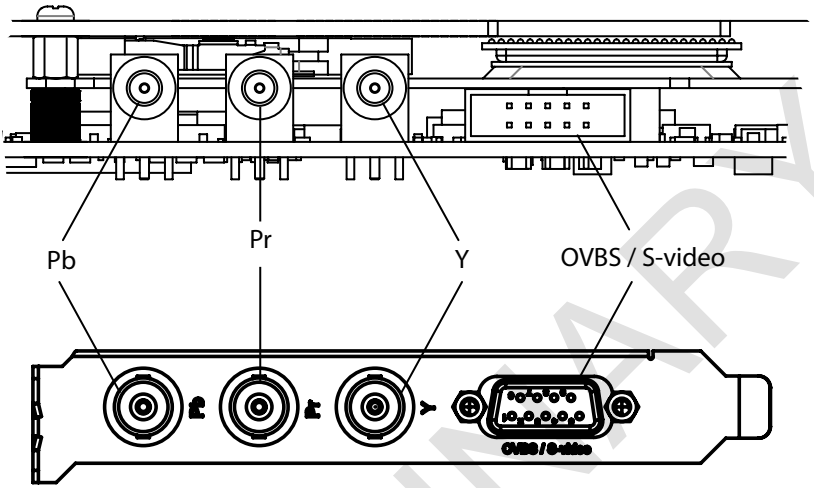


Figure 1-13: PCB-I/O Bracket Signal Correspondence

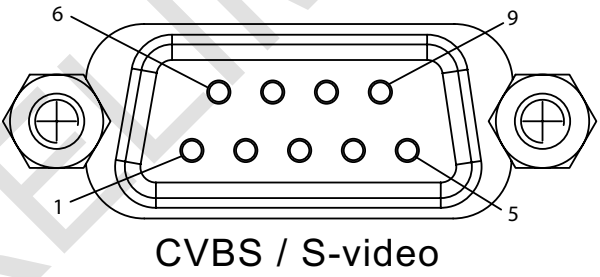


Figure 1-14: D-sub Connector for OVBS/S-video

Pin	Signal	In/Out
1	Y(S-Video)	Input
2	C(S-Video)	Input
3	Video Ground	

Pin	Signal	In/Out
4	N/A	
5	N/A	
6	CVBS	Input
7	Video Ground	
8	Video Ground	
9	N/A	

Table 1-9: D-sub Connector Pin Assignment

1.7.2 DVI-DVI/VGA Cable

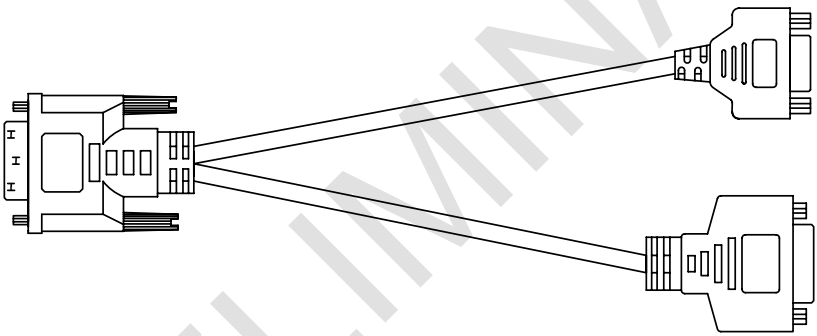


Figure 1-15: DVI-DVI/VGA Cable

DVI Connector

(for DVI connector diagram, See “DVI-I Connector” on page 8.

Pin	Signal	In/Out	Pin	Signal	In/Out
1	T.M.D.S data2-	Input	16	Hot plug detect	Output
2	T.M.D.S data2+	Input	17	T.M.D.S data0-	Input
3	T.M.D.S data2 shield		18	T.M.D.S data0+	Input
4	NC		19	T.M.D.S data2 shield	
5	NC		20	NC	
6	DDC clock	Input	21	NC	

Pin	Signal	In/Out	Pin	Signal	In/Out
7	DDC data	Input	22	T.M.D.S clock shield	
8	Analog Vertical Sync.	Input	23	T.M.D.S clock+	Input
9	T.M.D.S data1-	Input	24	T.M.D.S clock-	Input
10	T.M.D.S data1+	Input			
11	T.M.D.S data1 shield		C1	Analog Red	Input
12	NC		C2	Analog Green	Input
13	NC		C3	Analog Blue	Input
14	+5V Power	Output	C4	Analog Horizontal Snyc.	Input
15	Ground (Return for +5V,Hsync and Vsync)		C5	Analog Ground (Return for Analog R, G, and B)	

Table 1-10: DVI Connector Pin Assignment

VGA Connector

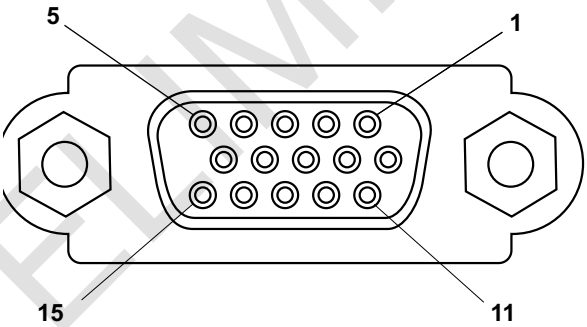


Figure 1-16: VGA connector for DVI-DVI/VGA Cable

Pin	Signal	In/Out
1	Analog Red	Input
2	Analog Green	Input
3	Analog Blue	Input
4	N/C	
5	Ground for Horizontal and Vertical Sync	

Pin	Signal	In/Out
6	Analog Ground for Analog Red	Input
7	Analog Ground for Analog Green	Input
8	Analog Ground for Analog Blue	Input
9	+5V Power	Output
10	Ground for Horizontal and Vertical SyN/C	
11	N/C	
12	N/C	
13	Analog Horizontal Sync	Input
14	Analog Vertical Sync	Input
15	N/C	
16	N/C	

Table 1-11: VGA Connector Pin Assignment

1.7.3 Dsub9–Composite/S-video Cable

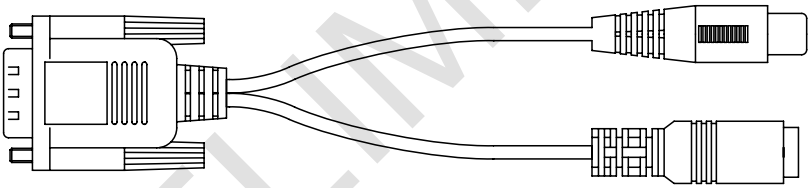


Figure 1-17: Dsub9-Composite/S-video Cable

2 Getting Started

2.1 Unpacking Checklist

Before unpacking, check the shipping carton for any damage. If the shipping carton and/or contents are damaged, inform your dealer immediately. Retain the shipping carton and packing materials for inspection. Obtain authorization from your dealer before returning any product to ADLINK. Ensure that the following items are included in the package.

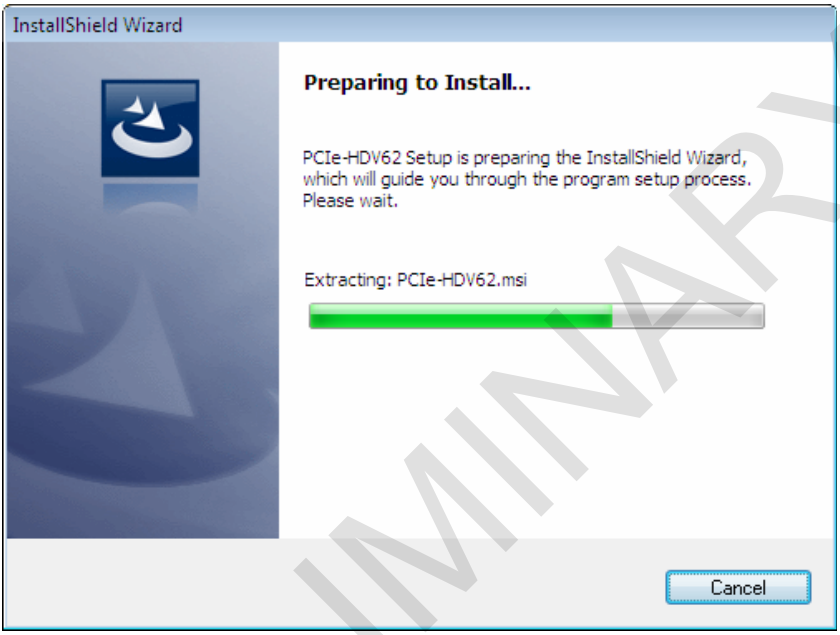
- HDV62A unit
- User's manual

OEM versions with non-standard configuration, functionality, or packaging may vary according to individual requirements.

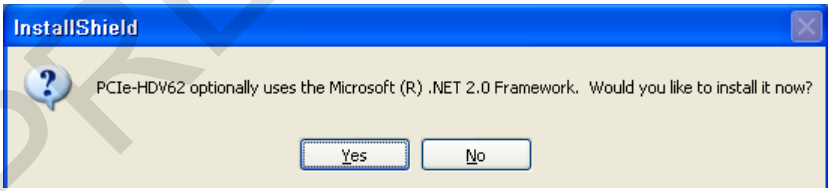
2.2 Windows Driver Installation

While the following driver installation procedure is demonstrated on a Windows Vista-equipped system, procedures for other Windows OSs are similar.

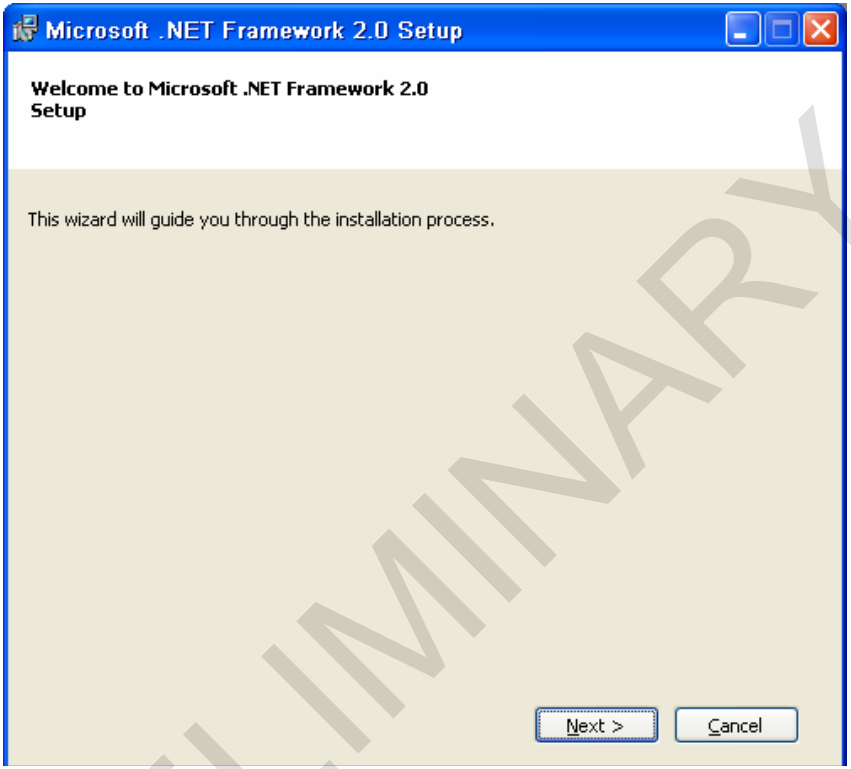
1. Run Setup. Installation commences



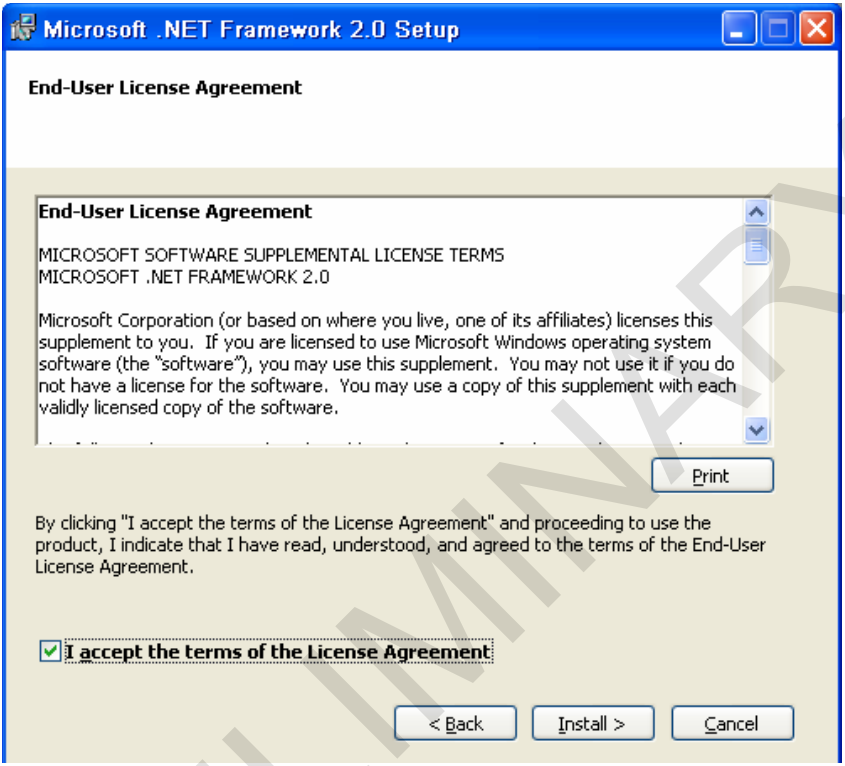
if .Net Framework 2.0 is not currently installed, confirmation of Installation appears.



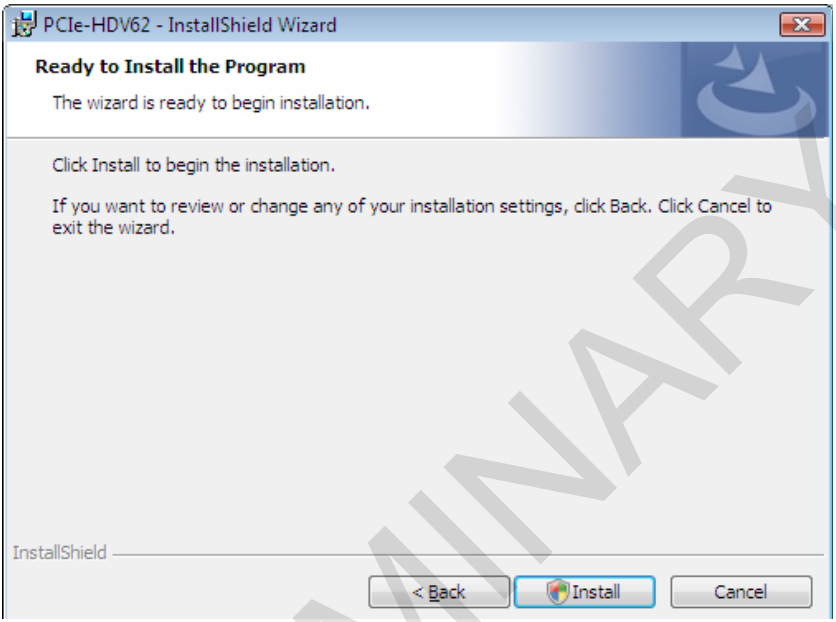
2. Select Yes to install .Net Framework.



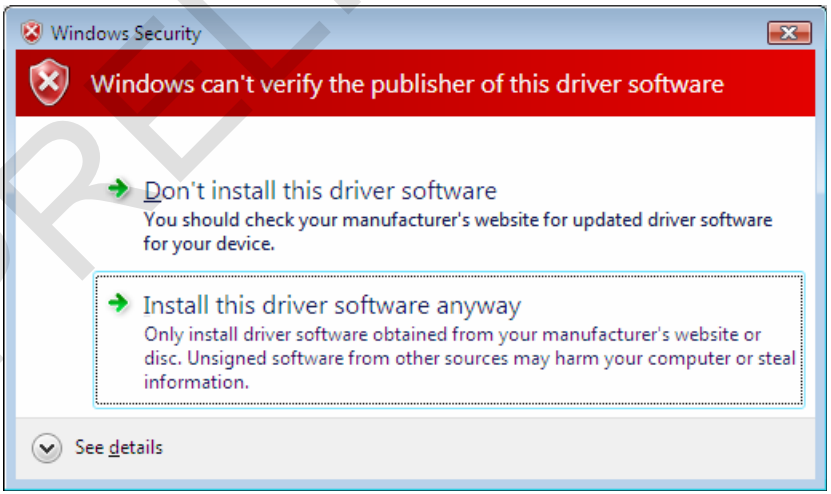
3. Accept the terms of the License Agreement and select Install when requested



4. Select Next until driver installation is completed.



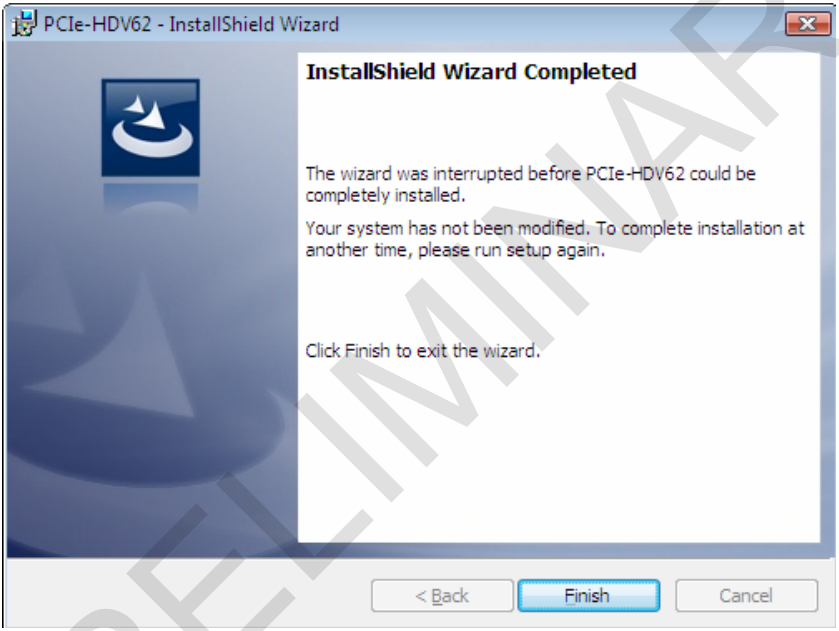
5. If a Windows Security warning appears, as shown, select “Install this driver anyway”.



NOTE:

If a “Found New Hardware Wizard” window appears, no action is required. Following installation, the window automatically closes.

6. If an installation failure notice appears as shown, select Finish

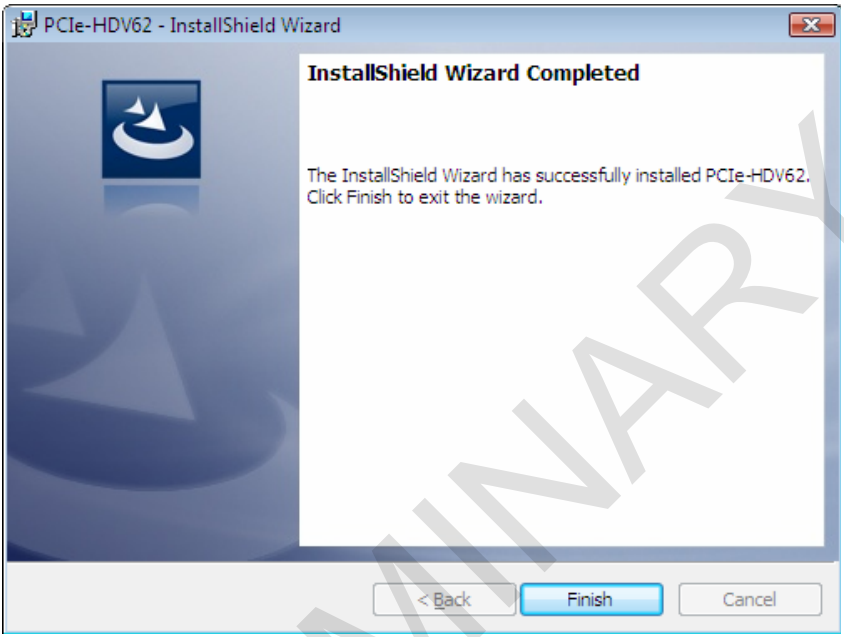


and please email the file setupapi.log (in the Windows folder) to ADLINK.

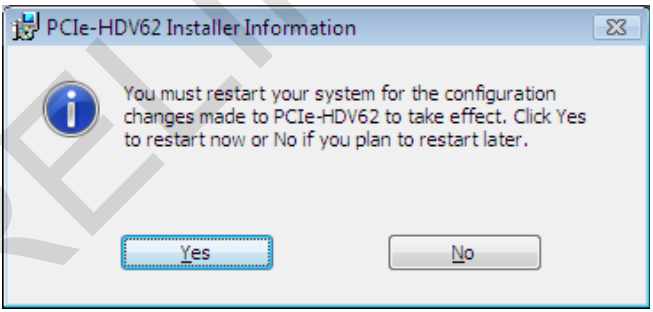
NOTE:

Log files on the Vista system are moved to %windir%\inf and renamed to setupapi.app.log and setupapi.dev.log where windir is the Windows folder.

7. If no error occurs, installation is complete, select Finish.

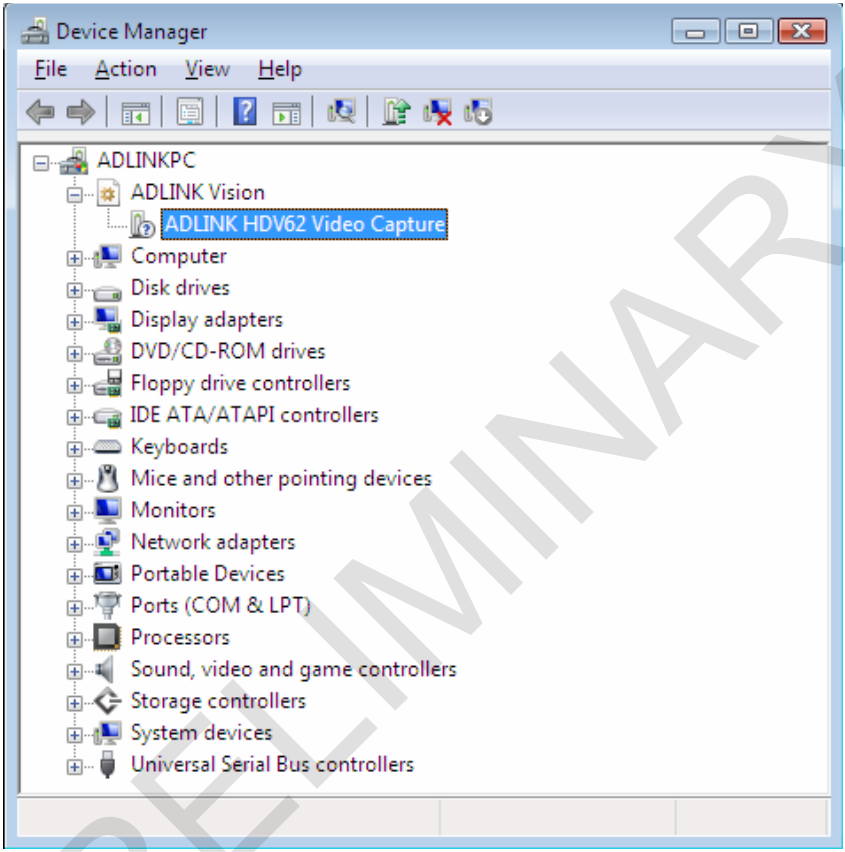


8. Select Yes to restart the system.



9. Open the Device Manager in the System directory of the Control Panel.

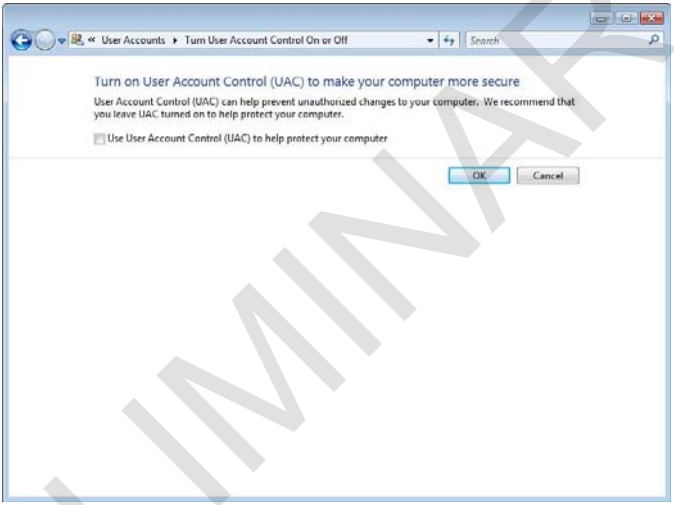
(If both HDV62 and HDV62A cards are installed, both will appear in the Device Manager)



.

NOTE:

For Windows Vista and Windows 7 users, security errors during operation of the ViewCreatorPro utility can be avoided by disabling User Account Control (UAC), at [Start] -> [Settings] -> [Control Panel] -> [User Accounts] -> [Turn User Account Control on or off]. Disable the UAC as shown, and restart the computer.



2.3 ViewCreator Pro Utility

The included ViewCreator Pro utility provides simple yet powerful setup, configuration, testing, and debugging of your vision system.

NOTE:

ViewCreator Pro is only available for Windows XP/Vista/7 systems, with recommended screen resolution exceeding 800x600

ViewCreator Pro provides

- 32/64-bit compatibility under Windows XP/Vista/7 Direct-Show driver
- Access to and configuration of HDV62A cards
- Video picture adjustment
- Image file (BMP or JPG) viewing and saving
- EDID R/W

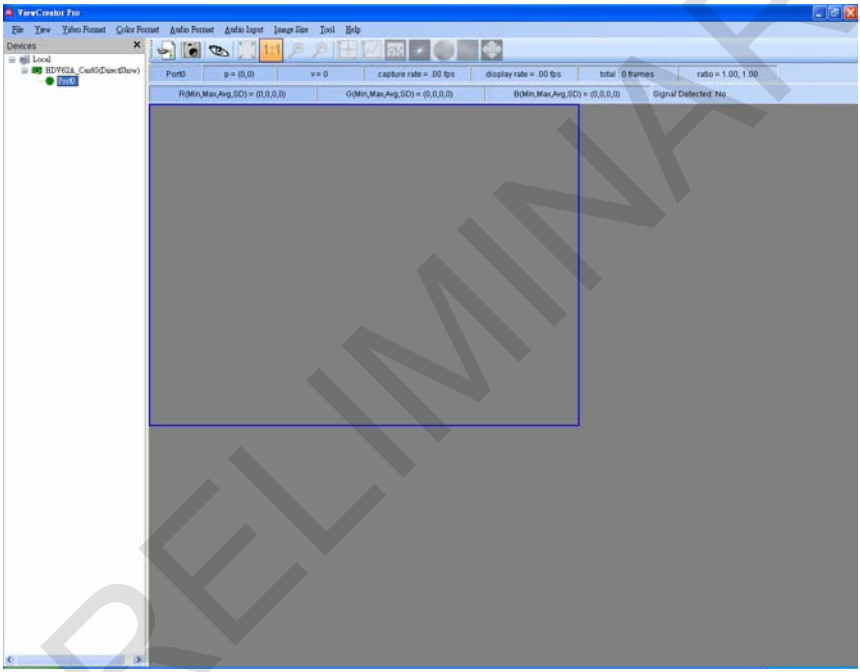


Figure 2-1: ViewCreator Pro Interface

The ViewCreator Pro interface provides a variety of panels and toolbars, allowing comprehensive task performance in all areas of function.

2.3.1 Devices Panel

Panel Item	Function
Local	Lists cards on the local system supported by ViewCreator Pro
Active Device	The device to which all operations will apply
Inactive Device	Currently inactive devices which can be activated by selection from this menu
Active port	The port to which all operations will apply
Inactive port	Currently inactive port which can be activated by selection from this menu

Table 2-1: Devices Panel Items

2.3.2 Adjustment Panel

Allows adjustment of image parameters including contrast, hue, saturation, and horizontal delay. Values can be entered by dragging the slider, or directly inputting a numeric value in the edit box. Contrast, hue, and saturation can be adjusted only when YPbPr format is selected.

Selection of Default returns all available parameters to factory-preset values.

2.3.3 Tool Panel

Tool	Button	Function
ContinueGrab		Toggles continuous acquisition of images
SnapShot		Captures a single image
Hide/Show Image		Toggles display of the image
FitSize		Resizes the image to fit the display area
OriginalSize		Restores the image to original size
ZoomIn		Increases image closeup
ZoomOut		Decreases image closeup
FocusCross		Displays blue crosshairs at any selected point on the image, the pixel values of which are displayed on the Status Panel
Focus Value		(See “FocusValue Operating Details” on page 31.)
Histogram		Opens a window showing the histogram of the display region
Fourier Transform		Opens a window showing the Fourier-Transformed image

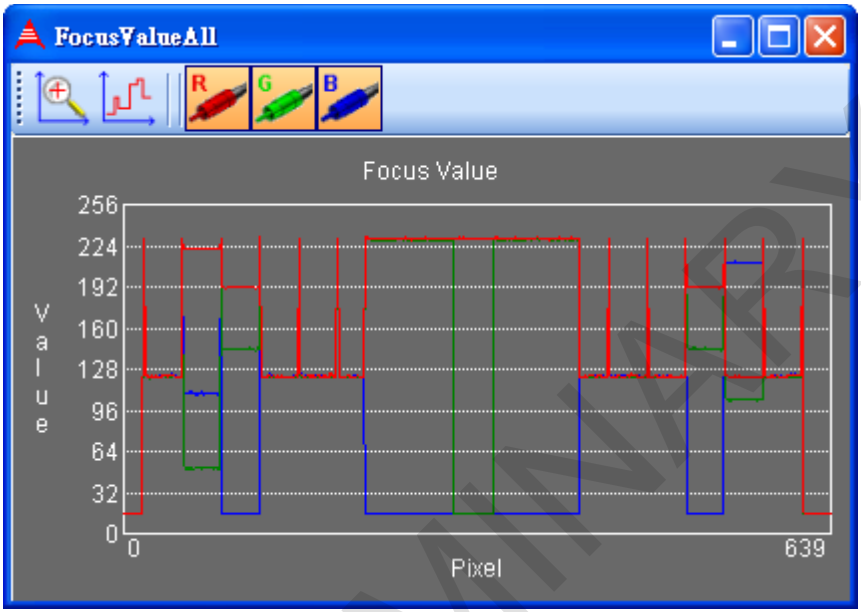
Tool	Button	Function
HSI Conversion		Opens a window showing the HSI values along the selected line on the image
YCbCr Enable		Transforms the image to YCbCr color space, toggling YCbCr Enable/Y Enable (CbCr=128)/ CbCr Enable (Y=0)/YCbCr Disable
Image Flip		Toggles No Flip/Flip X axis (horizontal)/Flip Y axis (vertical)

Table 2-2: Tool Panel Controls

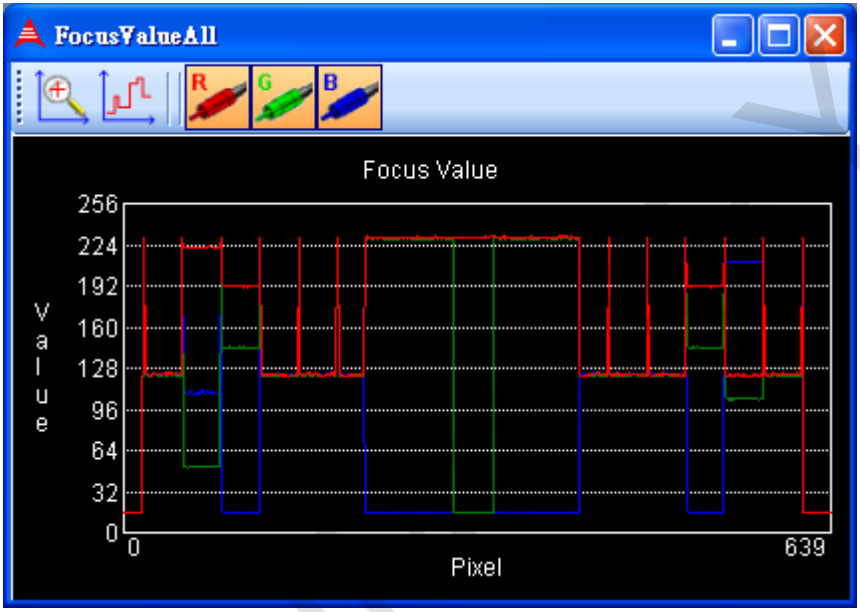
FocusValue Operating Details

A Focus Value window opens, displaying pixel values along a selected horizontal line of the image, which is displayed as red. Selecting the displayed image allows movement of the selected line.

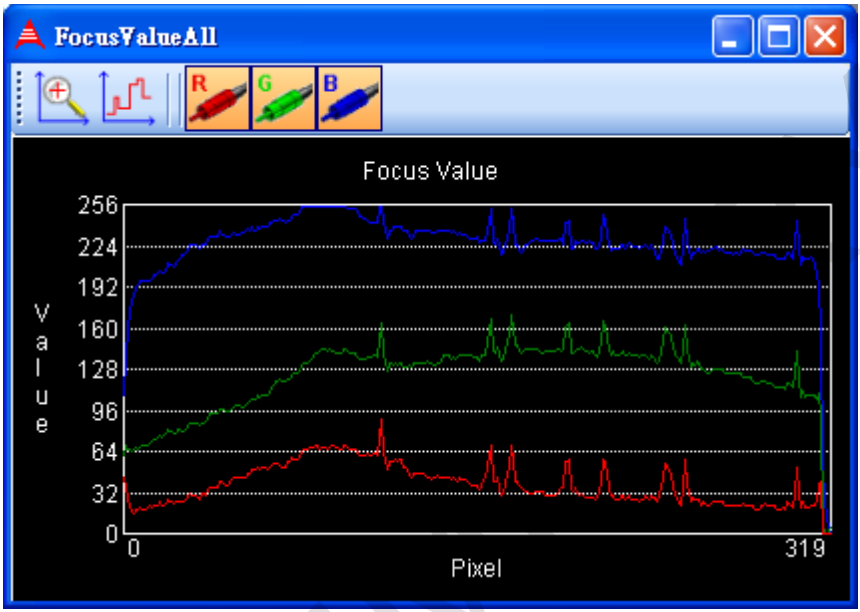
If acquisition is in progress, the background color of the window is gray. The chart updates immediately once an image is acquired. The x-axis region is determined by the number of horizontal pixels shown in the display panel.



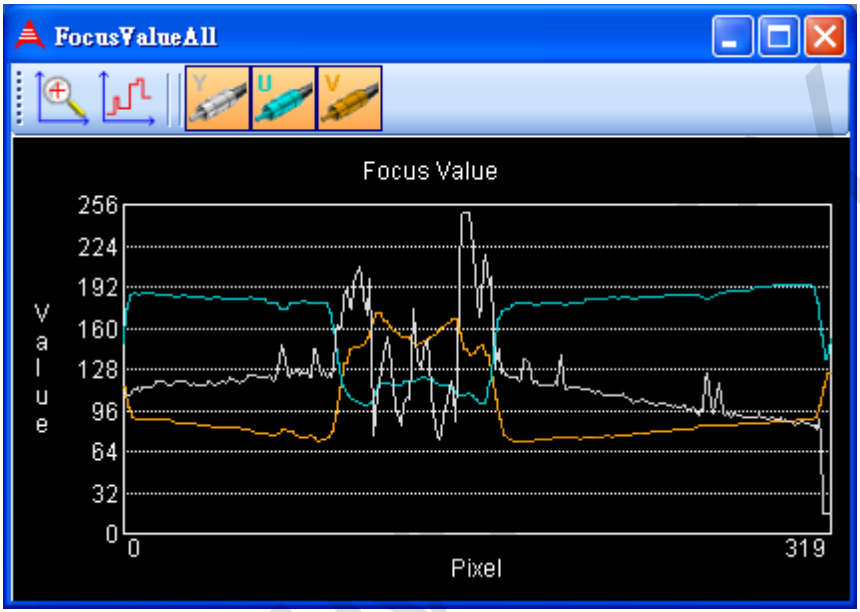
When acquisition is complete, the background color of the window changes to black. X-axis size is the width of the entire image.



If the image is chromatic, three curves, individually representing red, green, and blue are shown.



If the color format of captured image is YUV, three curves individually representing Y, U, and V are shown.



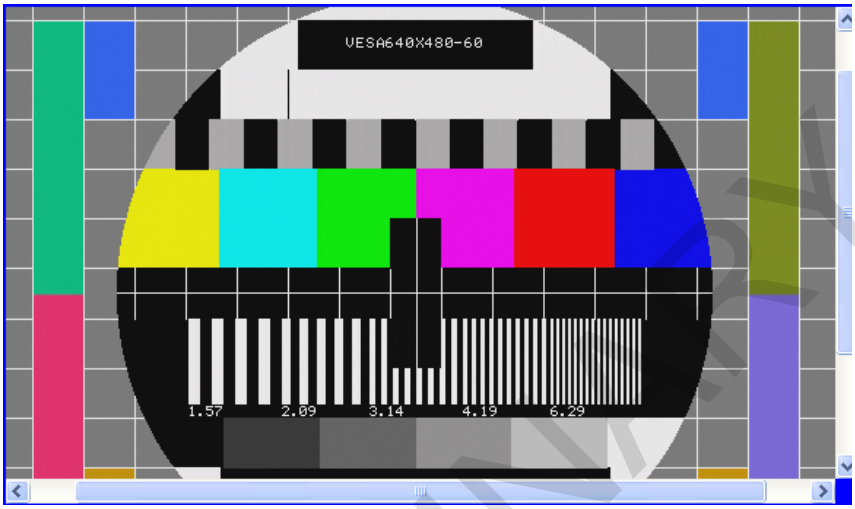
FocusValue Tools

Tool	Button	Function
Zoom In		Increases closeup on the area in the FocusValue window bound by the green rectangle, which can be resized by dragging the side borders
Differential		Displays slope of the line for the area in the FocusValue window bound by the green rectangle, which can be resized by dragging the side borders
Show/Hide Red Values		Toggles display of red pixel values.
Show/Hide Green Values		Toggles display of green pixel values.
Show/Hide Blue Values		Toggles display of blue pixel values.
Show/Hide Y Values		Toggles display of Y pixel values.
Show/Hide U Values		Toggles display of U pixel values.
Show/Hide V Values		Toggles display of V pixel values.
Focus Cross		Displays blue crosshairs at any selected point on the image, the pixel values of which are displayed on the Status Panel

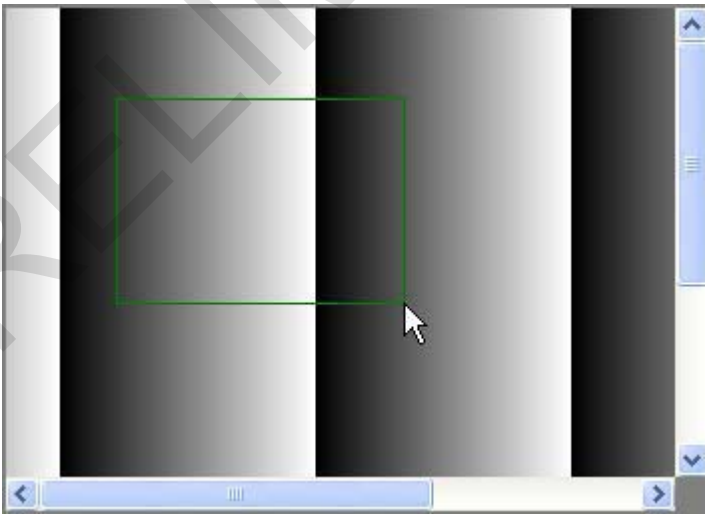
Table 2-3: FocusValue Tools

2.3.4 Display Panel

Captured images are displayed on this panel as shown.



Clicking and dragging the cursor generates a green rectangle, with the bound area magnified to the same size as the display region. Holding “Shift” while dragging retains the aspect ratio of the rectangle.



Right-clicking transforms the cursor into a move2D icon. When the size of the image exceeds that of the display panel, this icon allows the image to be dragged.

2.3.5 Display Menu

Allows selection of desired video format based on the video source, with support for RGB (DVI-I), YPbPr (EXT), HDMI (DVI-I), Composite (EXT), and S-Video (EXT).

The RGB menu is shown open as an example.

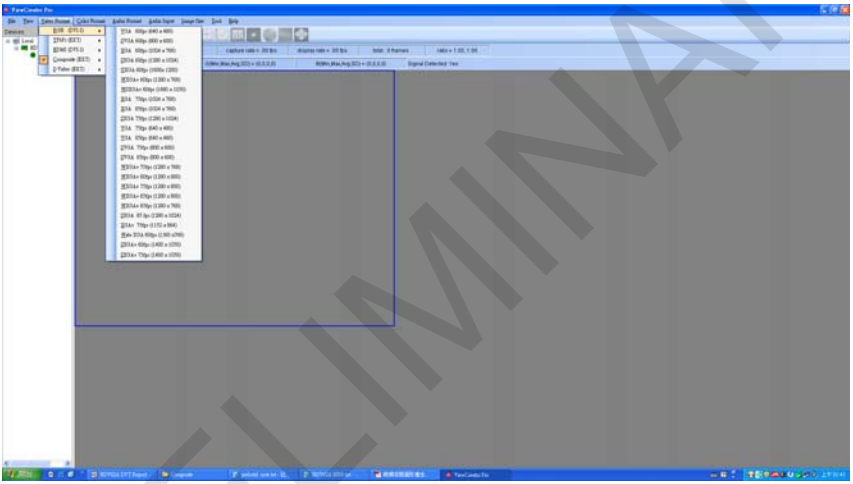


Figure 2-2: RGB Display Menu

RGB (DVI-I)

Format	Resolution
VGA 60 fps	640 x 480
SVGA 60 fps	800 x 600
XGA 60 fps	1024 x 768
SXGA 60 fps	1280 x 1024
UXGA 60fps	1600 x 1200
WXGA+ 60fps	1280 x 768
WSXGA+ 60fps	1680 x 1050
XGA 75fps	1024 x 768
XGA 85fps	1024 x 768
SXGA 75fps	1280 x 1024
VGA 75fps	640 x 480
VGA 85fps	640 x 480
SVGA 75fps	800 x 600
SVGA 85fps	800 x 600
WXGA+ 75fps	1280 x 768
WXGA+ 60fps	1280 x 800
WXGA+ 75fps	1280 x 800
WXGA+ 85fps	1280 x 800
WXGA+ 85fps	1280 x 768
SXGA 85 fps	1280 x 1024
XGA+ 75fps	1152 x 864
Wide XGA 60fps	1360 x768
SXGA+ 60fps	1400 x 1050
SXGA+ 75fps	1400 x 1050

Table 2-4: RGB (DVI-I) Available Formats

YPbPr (EXT)

Format	Resolution
525i 30 fps	720 x 480 interlace
625i 25 fps	720 x 576 interlace
525p 60 fps	720 x 480 progressive
625p 50 fps	720 x 576 progressive
720p 30 fps	1280 x 720 progressive
720p 50 fps	1280 x 720 progressive
720p 60 fps	1280 x 720 progressive
1080i 25 fps	1920 x 1080 interlace
1080i 30 fps	1920 x 1080 interlace
1080p 50 fps	1920 x 1080 progressive
1080p 60 fps	1920 x 1080 progressive

Table 2-5: YPbPr (EXT) Available Formats

HDMI (DVI-I)

Format	Resolution
720p 50 fps YCrCb In	1280 x 720 progressive
720p 60 fps YCrCb In	1280 x 720 progressive
1080i 25 fps YCrCb In	1920 x 1080 interlace
1080i 30 fps YCrCb In	1920 x 1080 interlace
1080p 25 fps YCrCb In	1920 x 1080 progressive
1080p 30 fps YCrCb In	1920 x 1080 progressive
1080p 50 fps YCrCb In	1920 x 1080 progressive
1080p 60 fps YCrCb In	1920 x 1080 progressive
VGA 60 fps	640 x 480
SVGA 60 fps	800 x 600
XGA 60 fps	1024 x 768
SXGA 60 fps	1280 x 1024
UXGA 60 fps	1600 x 1200
720p 50 fps RGB In	1280 x 720 progressive
720p 60 fps RGB In	1280 x 720 progressive
1080i 25 fps RGB In	1920 x 1080 interlace
1080i 30 fps RGB In	1920 x 1080 interlace
1080p 25 fps RGB In	1920 x 1080 progressive
1080p 30 fps RGB In	1920 x 1080 progressive
1080p 50 fps RGB In	1920 x 1080 progressive
1080p 60 fps RGB In	1920 x 1080 progressive
1080p 24 fps YCrCb In	1920 x 1080 progressive
1080p 48 fps YCrCb In	1920 x 1080 progressive
WXGA 60 fps YCrCb In	1360 x 768
1200p 50 fps YCrCb In	1920 x 1200 progressive
1200p 60 fps YCrCb In	1920 x 1200 progressive
1080p 24 fps RGB In	1920 x 1080 progressive
1080p 48 fps RGB In	1920 x 1080 progressive
WXGA 60 fps RGB In	1360 x 768
1200p 50 fps RGB In	1920 x 120 progressive
1200p 60 fps RGB In	1920 x 1200 progressive

Format	Resolution
525i 30 fps YCrCb In	720 x 480 interlace
625i 25 fps YCrCb In	720 x 576 interlace
525i 30 fps RGB In	720 x 480 interlace
625i 25 fps RGB In	720 x 576 interlace
WXGA+ 60 fps RGB In	1280 x 768
WSXGA+ 60 fps RGB In	1680 x 1050
VGA 60 fps YCrCb In	640 x 480
SVGA 60 fps YCrCb In	800 x 600
XGA 60 fps YCrCb In	1024 x 768
WXGA+ 60 fps YCrCb In	1280 x 768
WXGA 60 fps YCrCb In	1280 x 800
SXGA 60 fps YCrCb In	1280 x 1024
UXGA 60 fps YCrCb In	1600 x 1200
WSXGA+ 60 fps YCrCb In	1680 x 1050

Table 2-6: HDMI (DVI-I) Available Formats

Composite (EXT)

Format
NTSC-M
NTSC-J
PAL 60
NTSC 4.43
PAL BGHID
PAL N BGHID without pedestal)
PAL M without pedestal)
PAL M
PAL Nc
PAL Nc with pedestal)

Table 2-7: Composite (EXT) Available Formats

S-Video (EXT)

Format
NTSC-M
NTSC-J
PAL 60
NTSC 4.43
PAL BGHID
PAL N BGHID (without pedestal)
PAL M (without pedestal)
PAL M
PAL Nc
PAL Nc (with pedestal)

Table 2-8: S-Video (EXT) Available Formats

2.3.6 Color Format Menu

Sets the captured image to the selected color format.

2.3.7 Audio Format Menu

Sets the audio format, with available Sample Rates and Channels variable according to the input selected.

Sampling Rate	▷ 16 kHz ▷ 22 kHz ▷ 32 kHz ▷ 44 kHz ▷ 48 kHz ▷ 64 kHz ▷ 88 kHz ▷ 96 kHz ▷ 176 kHz ▷ 192 kHz
Sample Bits	24 bit

Channels	▷ 2.0 (2) ▷ 5.1 (6) ▷ 7.1 (8)

Table 2-9: Audio Formats

2.3.8 Audio Input Menu

Sets the audio input type.

2.3.9 Image Size Menu

Cropping

Before an image is cropped, a correct card index must first be selected. Sensor Width and Sensor Height are the width and height of the original image, varying according to the selected video format.

Parameters X, Y, Width, and Height must first be properly set, with X and Y the coordinates of the start position (upper left corner), and Width and Height the dimensions after cropping.

Selecting OK applies changes, and selecting Continue Grab on the Tool panel following cropping acquires the cropped images.

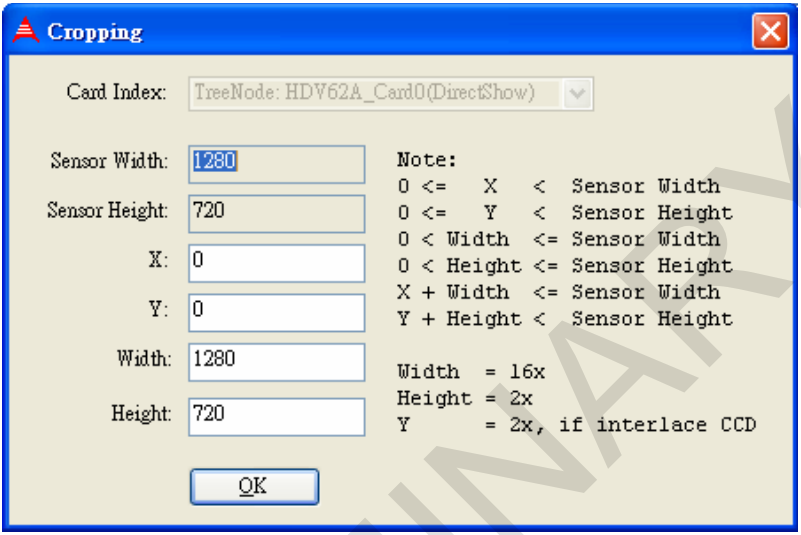


Figure 2-3: Cropping Dialog

Top-down

Enables top-down image display

Auto Detect

Automatically detects image resolution

2.3.10 Tool Menu

EDID (extended display identification data)

Before using EDID, a corresponding card index must be selected. After entering the offset and value, click the Write or Read button to write to or read from the EDID ROM. Valid range for the offset and value is 0 to 255. The values in EDID ROM are saved when the system is powered off.

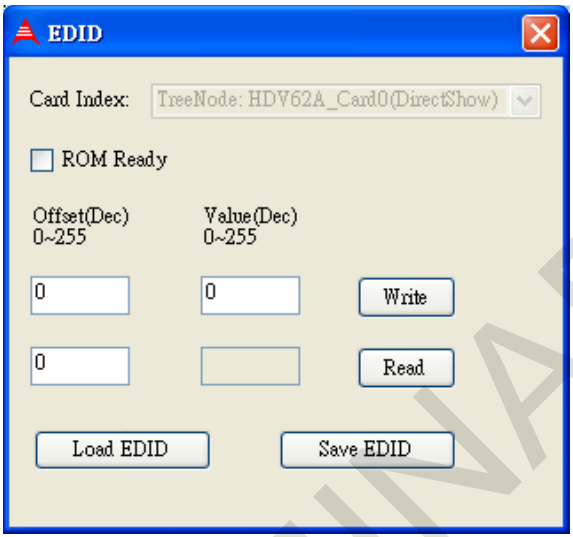


Figure 2-4: EDID Dialog

3 Function Library

This chapter describes the API (Application Programming Interface) for the HDV62A, for development of application programs within Visual C++, C#, Visual Basic.Net, Delphi, and Borland C++ Builder.

While the HDV62 API is based on DirectShow technologies, complexity of DirectShow programming has been eliminated in favor of simple API functions requiring no familiarity with DirectShow programming.

DirectShow technologies can alternativerly be utilized, as described in Chapter 4: DirectShow Programming Guide to program applications.

Please note that the API and DirectShow programming cannot be combined to access a single HDV62A card at the same time.

3.1 List of Functions

Category	Function Name
Device Control	Hdv62_GetDeviceCount
	Hdv62_DeviceOpen
	Hdv62_DeviceClose
	Hdv62_GetDeviceVendorName
	Hdv62_GetDeviceModelName
	Hdv62_GetDeviceVersion
	Hdv62_GetDeviceFirmwareVersion
	Hdv62_GetDriverVersion
	Hdv62_GetLibraryVersion
	Hdv62_GetDeviceID
	Hdv62_DeviceReset

Category	Function Name
Image Format Control	Hdv62_SetChannel
	Hdv62_GetChannel
	Hdv62_SetSensorFormat
	Hdv62_GetSensorFormat
	Hdv62_GetSensorWidth
	Hdv62_GetSensorHeight
	Hdv62_SetWidth
	Hdv62_GetWidth
	Hdv62_SetHeight
	Hdv62_GetHeight
	Hdv62_SetXOffset
	Hdv62_GetXOffset
	Hdv62_SetYOffset
	Hdv62_GetYOffset
	Hdv62_SetOutputFormat
	Hdv62_GetOutputFormat
	Hdv62_SetHDelay
	Hdv62_GetHDelay
	Hdv62_SetContrast
	Hdv62_GetContrast
	Hdv62_SetHue
	Hdv62_GetHue
	Hdv62_SetSaturation
	Hdv62_GetSaturation
	Hdv62_SetBrightness
	Hdv62_GetBrightness
	Hdv62_GetHdmiSensorResolution
	Hdv62_GetSDSensorResolution
	Hdv62_GetAnalogSensorResolution
	Hdv62_GetVideoCapabilities
	Hdv62_SetImageOrientation
	Hdv62_GetImageOrientation

Category	Function Name
Event & Callback	Hdv62_SetEventSelector
	Hdv62_GetEventSelector
	Hdv62_SetEventHandle
	Hdv62_GetEventHandle
	Hdv62_SetCallbackSelector
	Hdv62_GetCallbackSelector
	Hdv62_SetCallback
Acquisition Control	Hdv62_SetAcquisitionFrameCount
	Hdv62_GetAcquisitionFrameCount
	Hdv62_AcquisitionStart
	Hdv62_AcquisitionStop
	Hdv62_OneShot
	Hdv62_GetImageStream
	Hdv62_GetAcquisitionStatus
	Hdv62_GetAcquisitionStatistics
	Hdv62_GetSensorStatus
	Hdv62_SaveImage
	Hdv62_GetAudioStream
Others	Hdv62_GetErrorText
EDID	Hdv62_SetEdidReadyStatus
	Hdv62_GetEdidReadyStatus
	Hdv62_SetEdidAccessPermission
	Hdv62_GetEdidAccessPermission
	Hdv62_SetEdidWriteProtection
	Hdv62_GetEdidWriteProtection
	Hdv62_SetEdidRomSelector
	Hdv62_GetEdidRomSelector
	Hdv62_SetEdidRom
	Hdv62_GetEdidRom

Category	Function Name
Audio Format Control	Hdv62_SetAudioInput
	Hdv62_GetAudioInput
	Hdv62_SetAudioChannels
	Hdv62_GetAudioChannels
	Hdv62_SetAudioSamplesPerSec
	Hdv62_GetAudioSamplesPerSec
	Hdv62_SetAudioBitsPerSample
	Hdv62_GetAudioBitsPerSample
	Hdv62_GetAudioCapabilities

Table 3-1: API Functions

3.2 Setting Up the Build Environment

3.2.1 Include Files

All applications using API are required to include the following files.

Include File	
Hdv62.h	The header file required for all **C/C++** applications.
Hdv62.vb	The function definitions required for all **VB.Net** applications.
Hdv62.cs	The function definitions required for all **C#** applications.

3.2.2 Library Files

All **C/C++** applications using API require the following library files.

Library File	
Hdv62.lib	Exports API function definitions. Required for all Visual **C/C++** applications..
Hdv62_bcb.lib	Exports API function definitions. Required for all Borland C++ Builder applications.

3.2.3 DLL Files

All applications using API require the following DLL files.

.

DLL File	
Hdv62.dll	Dynamic link library. Required for all applications.

All files are located in the directory \ADLINK\hdv62\Include

3.3 Device Control Functions

3.3.1 Device Count

Returns the total number of HDV62A devices in the system, with a maximum of 16 devices detectable.

Syntax

C/C++

```
int Hdv62_GetDeviceCount(UINT &Count)
```

C#

```
int GetDeviceCount(out uint Count)
```

VB.Net

```
GetDeviceCount (ByRef Count as UInteger) As
Integer
```

Parameter(s)

Count

The total number of devices installed.

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

3.3.2 Device Open

Initializes a specified device. Should be called before other functions except those with no Number parameter.

Syntax

C/C++

```
int Hdv62_DeviceOpen (UINT Number)
```

C#

```
int DeviceOpen (uint Number)
```

VB.Net

```
DeviceOpen (ByVal Number As UInteger) As Inte-
ger
```

Parameter(s)

Number

The number of the device to be opened, with allowed values from 0 to 15.

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

3.3.3 Device Close

Closes the device and releases all allocated resources, this function should be called before terminating the application.

Syntax

C/C++

```
int Hdv62_DeviceClose (UINT Number)
```

C#

```
int DeviceClose (uint Number)
```

VB.Net

```
DeviceClose (ByVal Number As UInteger) As
Integer
```

Parameter(s)

Number

The number of the device to be closed, with allowed values from 0 to 15.

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

3.3.4 Device Vendor Name

Retrieves or returns the vendor name.

Syntax

C/C++

```
int Hdv62_GetDeviceVendorName (char *Name)
```

C#

```
string GetDeviceVendorName ()
```

VB.Net

```
GetDeviceVendorName ()As String
```

Parameter(s)

Name

Points to a user-allocated buffer into which the function copies the vendor name string, such as “ADLINK”. The name is NULL-terminated.

Return Value

C/C++

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

C#

Return vendor name.

VB.Net

Return vendor name.

3.3.5 Device Model Name

Retrieves or returns the model name.

Syntax

C/C++

```
int  Hdv62_GetDeviceModelName  (UINT  Number,
char *Name)
```

C#

```
string GetDeviceModelName (unit Number)
```

VB.Net

```
GetDeviceModelName (ByVal Number as UInteger)
As String
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

Name

Points to a user-allocated buffer into which the function copies the model name string, for example, “HDV62”. The name is NULL-terminated.

Return Value

C/C++

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

C#

Return model name.

VB.Net

Return model name.

3.3.6 Device Version

Retrieves or returns the version of the hardware device.

Syntax

C/C++

```
int Hdv62_GetDeviceVersion (UINT Number, char
*Version)
```

C#

```
string GetDeviceVersion (uint Number)
```

VB.Net

```
GetDeviceVersion (ByVal Number as UInteger) As
String
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

Version

Points to a user-allocated buffer into which the function copies the version string. The version is NULL-terminated.

"A2/A1" is for carrier board plus daughter board configurations, with the former carrier board version and latter daughter board version.

"A2" is for single board configuration.

Return Value

C/C++

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

C#

Return version string.

VB.Net

Return version string.

3.3.7 Device Firmware Version

Retrieves or returns the version of the firmware.

Syntax

C/C++

```
int Hdv62_GetDeviceFirmwareVersion (UINT Number, char *Version)
```

C#

```
string GetDeviceFirmwareVersion (uint Number)
```

VB.Net

```
GetDeviceFirmwareVersion (ByVal Number as UInteger) As String
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

Version

Points to a user-allocated buffer into which the version string is entered in a “Year/Month/Day Hour:Minute” format, such as “2009/11/19 14:22”. The version is NULL-terminated.

Return Value

C/C++

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

C#

Return version string.

VB.Net

Return version string.

3.3.8 Driver Version

Retrieves or returns the driver version.

Syntax

C/C++

```
int Hdv62_GetDriverVersion (UINT Number, char *Version)
```

C#

```
string GetDriverVersion (uint Number)
```

VB.Net

```
GetDriverVersion (ByVal Number as UInteger) As String
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

Version

Points to a user-allocated buffer into which the version string is entered, such as “1.0.0.0”. The version is NULL-terminated.

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

C#

Return version string.

VB.Net

Return version string.

3.3.9 Library Version

Retrieves or returns the library version.

Syntax

C/C++

```
int Hdv62_GetLibraryVersion (UINT Number, char
*Version)
```

C#

```
string GetLibraryVersion (uint Number)
```

VB.Net

```
GetLibraryVersion (ByVal Number as UInteger)
As String
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

Version

Points to a user-allocated buffer into which the version string is entered, such as “1.0.0.0”. The version is NULL-terminated.

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

C#

Return version string.

VB.Net

Return version string.

3.3.10 Device ID

Acquires device card ID.

Syntax

C/C++

```
int Hdv62_GetDeviceID (UINT Number, UINT& ID)
```

C#

```
int GetDeviceID (uint Number, out uint ID)
```

VB.Net

```
GetDeviceID (ByVal Number as UInteger, ByRef
ID as UInteger) As Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

ID

Card ID can be set by DIP Switch on the card, with possible values from 0 to 15. Card ID can distinguish cards when multiple cards are installed, with different number settings as shown in Section 1.6.1: Card ID Switch (SW3). Default card ID ignores multiples.

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

3.3.11 Device Reset

Restores the HDV62A to its initial boot state, after which all settings must be reconfigured. Call only when the device behaves abnormally and will not restore. The effect of this function is essentiually that of a reboot in less time.

Syntax

C/C++

```
int Hdv62_DeviceReset (UINT Number)
```

C#

```
int DeviceReset (uint Number)
```

VB.Net

```
DeviceReset (ByVal Number as UInteger) As
Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

3.4 Image Format Control Functions

3.4.1 Channel

Sets or retrieves channel of device, based on multi-source inputs, only one of which is available at one time. The desired channel should be selected. Please see

Section 1.5: Schematics & Connections for source input connections.

Syntax

C/C++

```
int Hdv62_SetChannel (UINT Number, UINT Channel)
int Hdv62_GetChannel (UINT Number, UINT &Channel)
```

C#

```
int SetChannel (uint Number, uint Channel)
int GetChannel (uint Number, out uint Channel)
```

VB.Net

```
SetChannel (ByVal Number as UInteger, ByVal Channel as UInteger) As Integer
GetChannel (ByVal Number as UInteger, ByRef Channel as UInteger) As Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

Channel

Input Source, with 5 allowed values from 0 to 4, for:

0. Analog RGB signal from DVI-I connector
1. YPbPr signal from external I/O bracket
2. HDMI signal from DVI-I connector
3. Composite signal from external I/O bracket
4. S-Video signal from DVI-I connector

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Section 3.7 Other Functions for return code error information.

3.4.2 Sensor Format

Sets or retrieves image format of source input, with format differing according to the input channel.

Syntax

C/C++

```
int Hdv62_SetSensorFormat (UINT Number, UINT Format)
int Hdv62_GetSensorFormat (UINT Number, UINT &Format)
```

C#

```
int SetSensorFormat (uint Number, uint Format)
int GetSensorFormat (uint Number, out uint Format)
```

VB.Net

```
SetSensorFormat (ByVal Number as UInteger, ByVal Format as UInteger) As Integer
GetSensorFormat (ByVal Number as UInteger, ByRef Format as UInteger) As Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

Format

Source input format, with values differing by channel as:

Channel 0: Analog RGB signal from DVI-I connector

0: VGA 60 fps (640 x 480)

1: SVGA 60 fps (800 x 600)

2: XVGA 60 fps (1024 x 768)

3: SXVGA 60 fps (1280 x 1024)

4 : UXGA 60fps (1600 x 1200)

5 : WXGA+ 60fps (1280 x 768)

6 : WSXGA+ 60fps (1680 x 1050)

7 : XGA 75fps (1024 x 768)

8 : XGA 85fps (1024 x 768)

9 : SXGA 75fps (1280 x 1024)

10 : VGA 75fps (640 x 480)

11 : VGA 85fps (640 x 480)

12 : SVGA 75fps (800 x 600)

13 : SVGA 85fps (800 x 600)

14 : WXGA+ 75fps (1280 x 768)

15 : WXGA+ 60fps (1280 x 800)

16 : WXGA+ 75fps (1280 x 800)

17 : WXGA+ 85fps (1280 x 800)

18 : WXGA+ 85fps (1280 x 768)

19: SXGA 85 fps (1280 x 1024)

20 : XGA+ 75fps (1152 x 864)

21 : Wide XGA 60fps (1360 x768)

22 : SXGA+ 60fps (1400 x 1050)

23 : SXGA+ 75fps (1400 x 1050)

Channel 1: YPbPr signal from external I/O bracket

0: 525i 30 fps (720 x 480 interlace, in frames per second)

1: 625i 25 fps (720 x 576 interlace, in frames per second)

2: 525p 60 fps (720 x 480 progressive)

3: 625p 50 fps (720 x 576 progressive)

5: 720p 50 fps (1280 x 720 progressive)

6: 720p 60 fps (1280 x 720 progressive)

7: 1080i 25 fps (1920 x 1080 interlace, in frames per second)

8: 1080i 30 fps (1920 x 1080 interlace, in frames per second)

9: 1080p 50 fps (1920 x 1080 progressive)

10: 1080p 60 fps (1920 x 1080 progressive)

Channel 2: HDMI signal from DVI-I connector

0: 720p 50 fps YCrCb In (1280 x 720 progressive)

1: 720p 60 fps YCrCb In (1280 x 720 progressive),

2: 1080i 25 fps YCrCb In (1920 x 1080 interlace, in frames per second),

3: 1080i 30 fps YCrCb In (1920 x 1080 interlace, in frames per second),

4: 1080p 25 fps YCrCb In (1920 x 1080 progressive)

5: 1080p 30 fps YCrCb In (1920 x 1080 progressive)

6: 1080p 50 fps YCrCb In (1920 x 1080 progressive)

7: 1080p 60 fps YCrCb In (1920 x 1080 progressive)

8: VGA 60 fps (640 x 480)

9: SVGA 60 fps (800 x 600)

10: XGA 60 fps (1024 x 768)

11: SXGA 60 fps (1280 x 1024)

12: UXGA 60 fps (1600 x 1200)

13: 720p 50 fps RGB In (1280 x 720 progressive)

14: 720p 60 fps RGB In (1280 x 720 progressive)

15: 1080i 25 fps RGB In (1920 x 1080 interlace, in frames per second)

16: 1080i 30 fps RGB In (1920 x 1080 interlace, in frames per second)

17: 1080p 25 fps RGB In (1920 x 1080 progressive)

18: 1080p 30 fps RGB In (1920 x 1080 progressive)

19: 1080p 50 fps RGB In (1920 x 1080 progressive)

20: 1080p 60 fps RGB In (1920 x 1080 progressive)

21: 1080p 24 fps YCrCb In (1920 x 1080 progressive)

22: 1080p 48 fps YCrCb In (1920 x 1080 progressive)

23: WXGA 60 fps YCrCb In (1360 x 768)

24: 1200p 50 fps YCrCb In (1920 x 1200 progressive)

25: 1200p 60 fps YCrCb In (1920 x 1200 progressive)

26: 1080p 24 fps RGB In (1920 x 1080 progressive)

27: 1080p 48 fps RGB In (1920 x 1080 progressive)

28: WXGA 60 fps RGB In (1360 x 768)

29: 1200p 50 fps RGB In (1920 x 120 progressive)

30: 1200p 60 fps RGB In (1920 x 1200 progressive)

31: 525i 30 fps YCrCb In (720 x 480 interlace, in frame per second)

32: 625i 25 fps YCrCb In (720 x 576 interlace, in frame per second)

33: 525i 30 fps RGB In (720 x 480 interlace, in frame per second)

34: 625i 25 fps RGB In (720 x 576 interlace, in frame per second)

35: WXGA+ 60 fps RGB In (1280 x 768)

36: WSXGA+ 60 fps RGB In (1680 x 1050)

37: VGA 60 fps YCrCb In (640 x 480)

38: SVGA 60 fps YCrCb In (800 x 600)

39: XGA 60 fps YCrCb In (1024 x 768)

40: WXGA+ 60 fps YCrCb In (1280 x 768)

41: WXGA 60 fps YCrCb In (1280 x 800)

42: SXGA 60 fps YCrCb In (1280 x 1024)

43: UXGA 60 fps YCrCb In (1600 x 1200)

44: WSXGA+ 60 fps YCrCb In (1680 x 1050)

Channel 3: Composite from external I/O bracket

0: NTSC MJ

1: PAL 60

2: NTSC 4.43

3: PAL BGHID

4: PAL M

5: PAL Nc

Channel 4: S-Video from external I/O bracket

0: NTSC MJ

1: PAL 60

2: NTSC 4.43

3: PAL BGHID

4: PAL M

5: PAL Nc

NOTE:

- Final resolutions of channels 1,2,and 3 can be acquired with GetVideoCapabilities()
- Noise and black images lasting for a few seconds (actual time of noise and black images depends on the source) may occur if the input source is HDMI plus HDCP, and can be prevented by delaying commencing capture for a few seconds after initial connection of the HDMI device

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

3.4.3 Sensor Width

Acquires image width of the source input, the same value as shown in Sensor Format.

Syntax

C/C++

```
int Hdv62_GetSensorWidth (UINT Number, UINT &Width)
```

C#

```
int GetSensorWidth (uint Number, out uint Width)
```

VB.Net

```
GetSensorWidth (ByVal Number as UInteger, ByRef Width as UInteger) As Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

Width

The image width of source input as shown.

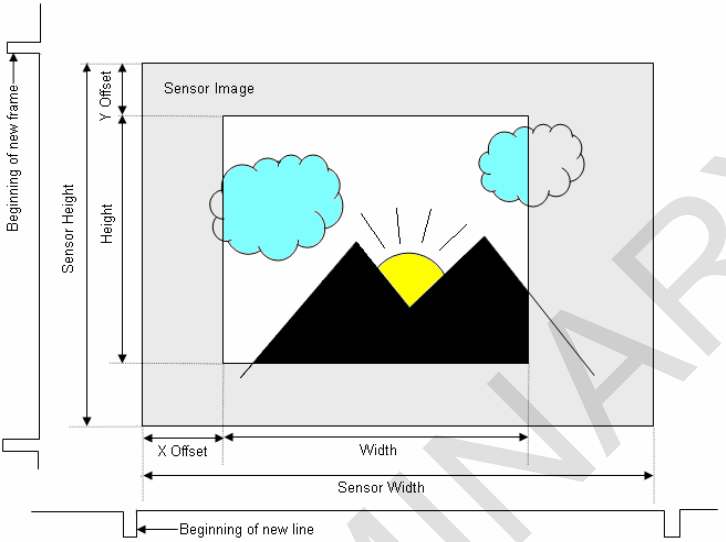


Figure 3-1: Image Layout

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

3.4.4 Sensor Height

Acquires image height of the source input, the same value as shown in Sensor Format.

Syntax

C/C++

```
int Hdv62_GetSensorHeight (UINT Number, UINT &
Height)
```

C#

```
int GetSensorHeight (uint Number, out uint
Height)
```

VB.Net

```
GetSensorHeight (ByVal Number as UInteger,
ByRef Height as UInteger) As Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

Height

The image height of source input as shown.

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

3.4.5 Width

Sets or retrieves active width of image, can reduce the number of horizontal pixels per line.

Syntax

C/C++

```
int Hdv62_SetWidth (UINT Number, UINT Width)
int Hdv62_GetWidth (UINT Number, UINT & Width)
```

C#

```
int SetWidth (uint Number, uint Width)
int GetWidth (uint Number, out uint Width)
```

VB.Net

```
SetWidth (ByVal Number as UInteger, ByVal
Width as UInteger) As Integer
GetWidth (ByVal Number as UInteger, ByRef
Width as UInteger) As Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

Width

Active width of the image as shown, the width coupled with XOffset acting as cropping parameters. The device crops the sensor image if the width is less than SensorWidth, according to:

Width ≦ SensorWidth

Width + XOffset ≦ SensorWidth

Width must be a multiple of 16 pixels

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

3.4.6 Height

Sets or retrieves active height of image, can reduce the number of lines.

Syntax

C/C++

```
int Hdv62_SetHeight (UINT Number, UINT Height)
int Hdv62_GetHeight (UINT Number, UINT& Height)
```

C#

```
int SetHeight (uint Number, uint Height)
int GetHeight (uint Number, out uint Height)
```

VB.Net

```
SetHeight (ByVal Number as UInteger, ByVal Height as UInteger) As Integer
GetHeight (ByVal Number as UInteger, ByRef Height as UInteger) As Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

Height

Active height of the image as shown, which when coupled with YOffset, acts as cropping parameters. The device crops the sensor image if the height is less than SensorHeight, according to:

Height <= SensorHeight

Height + YOffset <= SensorHeight

Height is an even number.

YOffset is an even number if sensor source is interlaced.

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

3.4.7 X Offset

Sets or retrieves X axis image cropping start lines.

Syntax

C/C++

```
int Hdv62_SetXOffset (UINT Number, UINT XOffset)
int Hdv62_GetXOffset (UINT Number, UINT & XOffset)
```

C#

```
int SetXOffset (uint Number, uint XOffset)
int GetXOffset (uint Number, out uint XOffset)
```

VB.Net

```
SetXOffset (ByVal Number as UInteger, ByVal XOffset as UInteger) As Integer
GetXOffset (ByVal Number as UInteger, ByRef XOffset as UInteger) As Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

XOffset

Start pixels of image cropping per line as shown, which, when coupled with Width, act as cropping parameters. The device crops sensor image if the XOffset exceeeds 0, according to:

Width <= SensorWidth

Width + XOffset <= SensorWidth

Width must be multiple of 16 pixels

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

3.4.8 Y Offset

Sets or retrieves Y axis image cropping start lines.

Syntax

C/C++

```
int Hdv62_SetYOffset (UINT Number, UINT YOff-
set)
int Hdv62_GetYOffset (UINT Number, UINT &
YOffset)
```

C#

```
int SetYOffset (uint Number, uint YOffset)
int GetYOffset (uint Number, out uint YOffset)
```

VB.Net

```
SetYOffset (ByVal Number as UInteger, ByVal
YOffset as UInteger) As Integer
GetYOffset (ByVal Number as UInteger, ByRef
YOffset as UInteger) As Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

YOffset

Image cropping start lines, as shown, which, coupled with Height, act as cropping parameters. The device crops sensor image if the YOffset exceeds 0, according to:

Height <= SensorHeight

Height + YOffset <= SensorHeight

Height is an even number

YOffset is an even number if sensor source is interlaced

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

3.4.9 Output Format

Sets or retrieves pixel output format.

Syntax

C/C++

```
int Hdv62_SetOutputFormat (UINT Number, UINT Format)
int Hdv62_GetOutputFormat (UINT Number, UINT & Format)
```

C#

```
int SetOutputFormat (uint Number, uint Format)
int GetOutputFormat (uint Number, out uint Format)
```

VB.Net

```
SetOutputFormat (ByVal Number as UInteger, ByVal Format as UInteger) As Integer
GetOutputFormat (ByVal Number as UInteger, ByRef Format as UInteger) As Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

Format

Pixel output format, one of (wherin x denotes neutral bit):

0: 24bit RGB (RGB24) – 8bit R + 8bit G + 8bit B

DWORD	Pixel Data [31:0]			
	[31:24]	[23:16]	[15:8]	[7:0]
dw0	B1	R0	G0	B0
dw1	G2	B2	R1	G1
dw2	R3	G3	B3	R2

1: 30bit RGB – 10bit R + 10bit G + 10bit B

DWORD	Pixel Data [31:0]			
	[31:24]	[23:16]	[15:8]	[7:0]
dw0	xx+B[9:4]	B[3:0]+G[9:6]	G[5:0]+R[9:8]	R[7:0]

2: 32bit RGB (RGB32) – 8bit R + 8bit G + 8bit B + 8bit Alpha

DWORD	Pixel Data [31:0]			
	[31:24]	[23:16]	[15:8]	[7:0]
dw0	Alpha	R	G	B

3: 8bit gray scale (Y8) – 8bit Y

DWORD	Pixel Data [31:0]			
	[31:24]	[23:16]	[15:8]	[7:0]
dw0	Y3	Y2	Y1	Y0

4: 24bit YCbCr 4:4:4 – 8bit Y + 8bit Cb + 8bit Cr

DWORD	Pixel Data [31:0]			
	[31:24]	[23:16]	[15:8]	[7:0]
dw0	Y1	Cr0	Cb0	Y0
dw1	Cb2	Y2	Cr1	Cb1
dw2	Cr3	Cb3	Y3	Cr2

5: 16bit YCbCr 4:2:2 (YUY2) – 8bit Y + 8bit Cb/Cr

DWORD	Pixel Data [31:0]			
	[31:24]	[23:16]	[15:8]	[7:0]
dw0	Cr	Y	Cb	Y

NOTE:

Actual width of the image setting in a WriteCropping routine must be multiple of 8 and actual height must be an even number

Total bytes of one scan line must align with a multiple of 16. If the requirement is not met, the HDV62A appends dummy bytes to the end of each line, for example, if 20bit YCbCr 4:2:2 is selected and:

- width = 800 pixels, each line = 2144 bytes (11 dummy bytes appended)
- width = 640 pixles, each line = 1712 bytes (5 dummy bytes appended)
- width = 720 pixles, each line = 1920 bytes (0 dummy bytes appended)

Formula: total bytes of each line = ((width * 16 / 6) + 15) & ~15 with all integer calculation

Formula of YCbCr to RGB:

- R = Y + 1.371(Cr-128)
- G = Y – 0.698(Cr-128) – 0.336(Cb-128)
- B = Y + 1.732(Cb-128)

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

3.4.10 HDelay

Sets or retrieves horizontal delay of frame images, similar to X offset, shifting images left or right to remove black vertical lines.

Syntax

C/C++

```
int Hdv62_SetHDelay (UINT Number, int Delay)
int Hdv62_GetHDelay (UINT Number, int & Delay)
```

C#

```
int SetHDelay (uint Number, int Delay)
```

```
int GetHDelay (uint Number, out int Delay)
```

VB.Net

```
SetHDelay (ByVal Number as UInteger, ByVal
Delay as Integer) As Integer
GetHDelay (ByVal Number as UInteger, ByRef
Delay as Integer) As Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

Delay

The horizontal delay of frame images, with allowed values from -3 to 3. Each resolution has a default value. Call this routine after channel and sensor format have been set.

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

3.4.11 Contrast

Sets or retrieves contrast of input YPbPr frame images.

Syntax

C/C++

```
int Hdv62_SetContrast (UINT Number, int Value)
int Hdv62_GetContrast (UINT Number, int &
Value)
```

C#

```
int SetContrast (uint Number, int Value)
int GetContrast (uint Number, out int Value)
```

VB.Net

```
SetContrast (ByVal Number as UInteger, ByVal
Value as Integer) As Integer
```

```
GetContrast (ByVal Number as UInteger, ByRef
Value as Integer) As Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

Value

Contrast of frame images, with allowed values from 0 to 255 and default value 128.

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

3.4.12 Hue

Sets or retrieves hue of input YPbPr frame images.

Syntax

C/C++

```
int Hdv62_SetHue (UINT Number, int Value)
int Hdv62_GetHue (UINT Number, int & Value)
```

C#

```
int SetHue (uint Number, int Value)
int GetHue (uint Number, out int Value)
```

VB.Net

```
SetHue (ByVal Number as UInteger, ByVal Value
as Integer) As Integer
GetHue (ByVal Number as UInteger, ByRef Value
as Integer) As Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

Value

Frame image hues, with allowed values from -128 to 127 and default value 0.

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

3.4.13 Saturation

Sets or retrieves saturation of input YPbPr frame images.

Syntax

C/C++

```
int Hdv62_SetSaturation (UINT Number, int Value)
int Hdv62_GetSaturation (UINT Number, int & Value)
```

C#

```
int SetSaturation (uint Number, int Value)
int GetSaturation (uint Number, out int Value)
```

VB.Net

```
SetSaturation (ByVal Number as UInteger, ByVal Value as Integer) As Integer
GetSaturation (ByVal Number as UInteger, ByRef Value as Integer) As Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

Value

Frame image saturation, with allowed values from 0 to 255 and default value 128.

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

3.4.14 Brightness

Sets or retrieves brightness of frame images.

Syntax

C/C++

```
int Hdv62_SetBrightness (UINT Number, int Value)
int Hdv62_GetBrightness (UINT Number, int & Value)
```

C#

```
int SetBrightness (uint Number, int Value)
int GetBrightness (uint Number, out int Value)
```

VB.Net

```
SetBrightness (ByVal Number as UInteger, ByVal Value as Integer) As Integer
GetBrightness (ByVal Number as UInteger, ByRef Value as Integer) As Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

Value

Brightness of frame images, with allowed values from -128 to 127 and default value 0.

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

3.4.15 HdmiSensorResolution

Retrieves the detected HDMI sensor resolution.

Syntax

C/C++

```
int   Hdv62_GetHdmiSensorResolution(UINT   Num-
ber, SENSOR_RESOLUTION& Resolution)
```

C#

```
int GetHdmiSensorResolution (uint Number, out
SENSOR_RESOLUTION Resolution)
```

VB.Net

```
GetHdmiSensorResolution (ByVal Number as UIn-
teger, ByRef Resolution As SENSOR_RESOLUTION)
As Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

SENSOR_RESOLUTION

Structure of HDMI sensor resolution defined as:

C/C++

typedef struct _SENSOR_RESOLUTION

{

UINT Width;

UINT Height;

UINT FrameRate;

UINT Interlace;

} SENSOR_RESOLUTION;

C#

public struct SENSOR_RESOLUTION

```
{
public uint Width;
public uint Height;
public uint FrameRate;
public uint Interlace;
}
```

VB.Net

```
Public Structure SENSOR_RESOLUTION
Public Width As UInteger
Public Height As UInteger
Public FrameRate As UInteger
Public Interlace As UInteger
End Structure
```

NOTE:

SD format (525i and 625i) may exceed the defined width, such as 625i being 720 x 576, but 1440 x 576 is usable from this function.

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

3.4.16 SDSensorResolution

Retrieves the detected resolution of the analog sensor including composite and S-Video signals. When the channel is set to one of the inputs described, after at least 1 second, the signal is detected correctly.

Syntax

C/C++

```
int  Hdv62_GetSDSensorResolution(UINT  Number,
int& Value)
```

C#

```
int GetSDSensorResolution (uint Number,  out
int Value)
```

VB.Net

```
GetSDSensorResolution (ByVal Number as UInte-
ger, ByRef Value As Integer) As Integer
```

Parameter(s)

Value

Read back resolution of the sensor, defined as:

0: NTSC MJ

1: PAL 60

2: NTSC 4.43

3: PAL BGHID

4: PAL M

5: PAL Nc

-1: Not detected

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

3.4.17 AnalogSensorResolution

Acquires the detected resolution of the analog sensor including analog RGB and YPbPr. When the channel is set to one of the decribed inputs, after least 1 second the signal can be detected correctly.

Syntax

C/C++

```
int Hdv62_GetAnalogSensorResolution(UINT Num-
ber, SENSOR_RESOLUTION& Resolution)
```

C#

```
int GetAnalogSensorResolution (uint Number,
out SENSOR_RESOLUTION Resolution)
```

VB.Net

```
GetAnalogSensorResolution (ByVal Number as
UInteger, ByRef Resolution As
SENSOR_RESOLUTION) As Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

SENSOR_RESOLUTION

Structure of analog sensor resolution, defined as:

C/C++

```
typedef struct _SENSOR_RESOLUTION
{
UINT Width;
UINT Height;
UINT FrameRate;
UINT Interlace;
} SENSOR_RESOLUTION;
```

C#

```
public struct SENSOR_RESOLUTION
{
public uint Width;
public uint Height;
```

public uint FrameRate;

public uint Interlace;

}

VB.Net

Public Structure SENSOR_RESOLUTION

Public Width As UInteger

Public Height As UInteger

Public FrameRate As UInteger

Public Interlace As UInteger

End Structure

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

3.4.18 Video Capabilities

Acquires list of resolutions the card supports.

Syntax

C/C++

```
int Hdv62_GetVideoCapabilities(UINT Number,
RESOLUTION_CAPABILITIES & Caps)
```

C#

```
int GetVideoCapabilities (uint Number, out
RESOLUTION_CAPABILITIES Caps)
```

VB.Net

```
GetVideoCapabilities (ByVal Number as UInte-
ger, ByRef Caps As RESOLUTION_CAPABILITIES) As
Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

RESOLUTION_CAPABILITIES

Supported resolutions defined as:

C/C++

```
typedef struct _SENSOR_PROPERTIES
{
   char Name[ 256 ];
   unsigned long Width;
   unsigned long Height;
   unsigned long FrameRate;
   unsigned long Interlace;
   unsigned long Hf;
   unsigned long HTotal;
   unsigned long Hsw;
   unsigned long Hbp;
   unsigned long Vf;
   unsigned long VTotal;
   unsigned long Vsw;
   unsigned long Vbp;
} SENSOR_PROPERTIES;

typedef struct _RESOLUTION_CAPABILITIES
{
   unsigned long NumRgbResolution;
   SENSOR_PROPERTIES *RgbResolutions;
```

```
    unsigned long NumYPbPrResolution;
    SENSOR_PROPERTIES *YPbPrResolutions;
    unsigned long NumHdmiResolution;
    SENSOR_PROPERTIES *HdmiResolutions;
} RESOLUTION_CAPABILITIES;
```

C#

```
struct SENSOR_PROPERTIES
{
    // name
MarshalAs(UnmanagedType.ByValTStr, SizeConst = 256)]
    public string Name;

    // video setting
    public uint Width;
    public uint Height;
    public uint FrameRate;
    public uint Interlace;

    // video timing
    public uint Hf;// horizontal frequency
    public uint HTotal;// horizontal total line
    public uint Hsw;// horizontal sync width
    public uint Hbp;// horizontal back porch
    public uint Vf;// vertical frequency
    public uint VTotal;// vertical total line
    public uint Vsw;// vertical sync width
    public uint Vbp;// vertical back porch
```

```
}

struct RESOLUTION_CAPABILITIES
{
   public uint NumRgbResolution;
   public IntPtr RgbResolutions;

   public uint NumYPbPrResolution;
   public IntPtr YPbPrResolutions;

   public uint NumHdmiResolution;
   public IntPtr HdmiResolutions;
}
```

VB.Net

```
Structure SENSOR_PROPERTIES

                <MarshalAs(UnmanagedType.ByValTStr,
SizeConst:=256)> _
   Dim Name As String
   Dim Width As UInteger
   Dim Height As UInteger
   Dim FrameRate As UInteger
   Dim Interlace As UInteger

   Dim Hf As UInteger      ' horizontal frequency
   Dim HTotal As UInteger  ' horizontal total line
   Dim Hsw As UInteger     ' horizontal sync width
```

```
    Dim Hbp As UInteger     ' horizontal back porch
    Dim Vf As UInteger      ' vertical frequency
    Dim VTotal As UInteger  ' vertical total line
    Dim Vsw As UInteger     ' vertical sync width
    Dim Vbp As UInteger     ' vertical back porch

End Structure

Structure RESOLUTION_CAPABILITIES

    Dim NumRgbResolution As UInteger
    Dim RgbResolutions As IntPtr

    Dim NumYPbPrResolution As UInteger
    Dim YPbPrResolutions As IntPtr

    Dim NumHdmiResolution As UInteger
    Dim HdmiResolutions As IntPtr

End Structure
```

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

3.4.19 Image Orientaton

Sets or retrieves orientation of images arranged in memory.

Syntax

C/C++

```
int  Hdv62_SetImageOrientation  (UINT  Number,
UINT Value)

int  Hdv62_GetImageOrientation  (UINT  Number,
UINT& Value)
```

C#

```
int  SetImageOrientation  (uint  Number,  uint
Value)

int GetImageOrientation (uint Number, out uint
Value)
```

VB.Net

```
SetImageOrientation (ByVal Number as UInteger,
ByVal Value As UInteger) As Integer

GetImageOrientation (ByVal Number as UInteger,
ByRef Value As UInteger) As Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

Value

Indicates the image orientation, from among:

0: Bottom-up, in which the image buffer starts with the bottom row of pixels, followed by the next row up, and so forth, with the top row of the image the last row in the buffer, such that the first byte in memory is the bottom-left pixel of the image. Physical layout of a bottom-up image is as shown.

E.g. Color space = RGB24

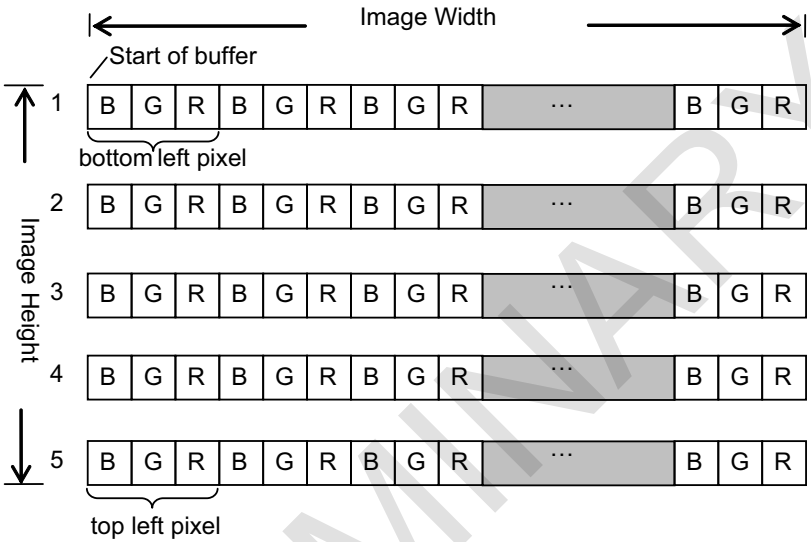


Figure 3-2: Bottom-up Image Orientation

1: Top-down, in which order of the row is reversed, with top row of the image the first row in memory, followed by the next row down, with bottom row of the image the last row in the buffer, such that first byte in memory is the top-left of the image. Physical layout of a top-down image is as shown.

E.g. Color space = RGB24

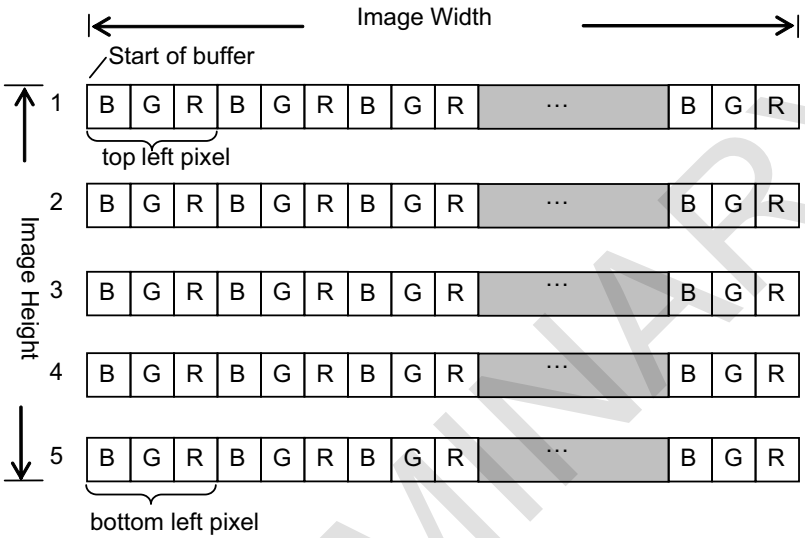


Figure 3-3: Top-down Image Orienation

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

3.5 Event and Callback Functions

3.5.1 Event Selector

Sets or retrieves the event type.

Syntax

C/C++

```
int Hdv62_SetEventSelector (UINT Number, UINT
Mode)
int Hdv62_GetEventSelector (UINT Number, UINT
& Mode)
```

C#

```
int SetEventSelector (uint Number, uint Mode)
int GetEventSelector (uint Number, out uint Mode)
```

VB.Net

```
SetEventSelector (ByVal Number as UInteger, ByVal Mode as UInteger) As Integer
GetEventSelector (ByVal Number as UInteger, ByRef Mode as UInteger) As Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

Mode

Event type, comprising frame and DI events, with frame event the result of a library issue of an event when a frame is ready, and DI event the result of library issue of an event when the state of any DI has changed, wherein mode can be:

0: Frame event

2: Audio event

And SetEventHandle sets an event handle.

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

3.5.2 Event Handle

Sets or retrieves event handle. While event and callback are the most frequently used methods determining when to recover frame or DI state information, only one is normally allowed, but here both can be set.

Syntax

C/C++

```
int Hdv62_SetEventHandle (UINT Number, HANDLE Handle)

int Hdv62_GetEventHandle (UINT Number, HANDLE &Handle)
```

C#

```
int SetEventHandle (uint Number, IntPtr Handle)

int SetEventHandle (uint Number, SafeWaitHandle Handle)

int GetEventHandle (uint Number, out IntPtr Handle)

int GetEventHandle (uint Number, out SafeWaitHandle Handle)
```

VB.Net

```
SetEventHandle (ByVal Number as UInteger, ByVal Handle as IntPtr) As Integer

SetEventHandle (ByVal Number as U Integer, ByVal Handle As SafeWaitHandle) As Integer

GetEventHandle (ByVal Number as UInteger, ByRef Handle as IntPtr) As Integer

GetEventHandle (ByVal Number as UInteger, ByRef Handle As SafeWaitHandle) As Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

Handle

Event handle created by the application. After the application waits for an event, call:

- ▷ **GetImageStream** to acquire the pointer of the frame buffer
- ▷ **GetDI** to acquire DI state
- ▷ **GetAudioStream** to acquire audi data pointer and size

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

3.5.3 CallbackSelector

Sets or retrieves callback function type.

Syntax

C/C++

```
int  Hdv62_SetCallbackSelector  (UINT  Number,
UINT Mode)
int  Hdv62_GetCallbackSelector  (UINT  Number,
UINT Mode)
```

C#

```
int  SetCallbackSelector  (uint  Number,  uint
Mode)
int GetCallbackSelector (uint Number, out uint
Mode)
```

VB.Net

```
SetCallbackSelector (ByVal Number as UInteger,
ByVal Mode as UInteger) As Integer
GetCallbackSelector (ByVal Number as UInteger,
ByRef Mode as UInteger) As Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

Mode

Callback type, from among frame callback,in which the library calls the callback routine when a frame is ready, or audio callback. **SetCallback** sets a callback function. Mod can be:

0: Frame callback

2: Audio callback

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

3.5.4 Callback

Sets or retreives callback handle. While event and callback are the most frequently used methods determining when to recover frame or DI state information, only one is normally allowed, but here both can be set.

Syntax

C/C++

```
int Hdv62_SetCallback (UINT Number, HDV62CALLBACK Fun)
```

C#

```
int SetCallback (uint Number, HDV62CALLBACK Fun)
```

VB.Net

```
SetCallback (ByVal Number as UInteger, ByVal Fun as HDV62CALLBACK) As Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

Fun

Pointer for callback routine, in which a callback function must be declared and set as the parameter. In the callback function, call:

- ▷ **GetImageStream** to acquire the pointer of the frame buffer
- ▷ **GetDI** to acquire DI state
- ▷ **GetAudioStream** to acquire audi data pointer and size

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

3.6 Acquisition Control Functions

3.6.1 Acquisition Frame Count

Sets or retrieves the number of frames to be simulatane-ously captured. Call:

- ▷ **AcquisitionStart** to initiate capture
- ▷ **GetAcquisitionStatus** to retreive acquisition state
- ▷ **AcquisitionStop** to terminate capture

Syntax

C/C++

```
int Hdv62_SetAcquisitionFrameCount (UINT Number, UINT Count)
int Hdv62_GetAcquisitionFrameCount (UINT Number, UINT & Count)
```

C#

```
int SetAcquisitionFrameCount (uint Number, uint Count)
int GetAcquisitionFrameCount (uint Number, out uint Count)
```

VB.Net

```
SetAcquisitionFrameCount (ByVal Number as UIn-
teger, ByVal Count as UInteger) As Integer

GetAcquisitionFrameCount (ByVal Number as UIn-
teger, ByRef Count as UInteger) As Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

Count

The frame count to be captured, from among:

- ▷ 0: Capture until AcquisitionStop is called.
- ▷ >0: Acquires the desired frame count, and, when reached, acquisition status is changed to 0 (stopped). AcquisitionStop must be called to stop acquisition.

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

3.6.2 Acquisition Start

Initiates frame capture

Syntax

C/C++

```
int Hdv62_AcquisitionStart (UINT Number)
```

C#

```
int AcquisitionStart (uint Number)
```

VB.Net

```
AcquisitionStart (ByVal Number as UInteger) As
Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

Noise and black images lasting for a few seconds (actual time of noise and black images depends on the source) may occur if the input source is HDMI plus HDCP, and can be prevented by delaying commencing capture for a few seconds after initial connection of the HDMI device

3.6.3 Acquisition Stop

Terminates frame capture

Syntax

C/C++

```
int Hdv62_AcquisitionStop (UINT Number)
```

C#

```
int AcquisitionStop (uint Number)
```

VB.Net

```
AcquisitionStop (ByVal Number as UInteger) As
Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

3.6.4 One Shot

Acquires a single frame image within a specific time, as an independent function which cannot be used with AcquisitionStart, Callback, or Event. When complete without errors, calling GetImageStream and/or GetAudioStream retreives the frame image pointer.

Syntax

C/C++

```
int Hdv62_OneShot (UINT Number, UINT Timeout)
```

C#

```
int OneShot (uint Number, uint Timeout)
```

VB.Net

```
OneShot (ByVal Number as UInteger, ByVal Time-
out as UInteger) As Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

Timeout

Maximum waiting time for acquisition, in milliseconds.

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

3.6.5 Image Stream

Retreives the image buffer pointer. Usually called during callback, after waiting for a frame event, or after calling OneShot.

Syntax

C/C++

```
int Hdv62_GetImageStream (UINT Number, void
**Buffer)
```

C#

```
int GetImageStream (uint Number, out IntPtr
Buffer)
```

VB.Net

```
GetImageStream (ByVal Number as UInteger,
ByRef Buffer as IntPtr) As Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

Buffer

Image buffer pointer.

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

3.6.6 Acquisition Status

Retreives current status of the image acquisition.

Syntax

C/C++

```
int Hdv62_GetAcquisitionStatus (UINT Number,
UINT &Status)
```

C#

```
int GetAcquisitionStatus (uint Number, out
uint Status)
```

VB.Net

```
GetAcquisitionStatus (ByVal Number as UInte-
ger, ByRef Status as UInteger) As Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

Status

Acquisition status, from among:

0: Stopped

1: Running

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

3.6.7 Acquisition Statistics

Acquires the number of frames captured since Acquisition Start.

Syntax

C/C++

```
int Hdv62_GetAcquisitionStatistics (UINT Number, UINT &Count)
```

C#

```
int GetAcquisitionStatistics (uint Number, out uint Count)
```

VB.Net

```
GetAcquisitionStatistics (ByVal Number as UInteger, ByRef Count as UInteger) As Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

Count

Number of frames captured.

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

3.6.8 Sensor Status

Determines whether the device is connected to a suitable sensor.

Syntax

C/C++

```
int Hdv62_GetSensorStatus (UINT Number, UINT&
Locked)
```

C#

```
int GetSensorStatus (uint Number, out uint
Locked)
```

VB.Net

```
GetSensorStatus (ByVal Number as UInteger,
ByRef Locked as UInterger) As Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

Locked

In determining whether the input signal is locked, establishes connection of a suitable sensor, from among:

- ▷ 0: No proper signal is detected
- ▷ 1: A proper signal is detected


NOTE:

- ▶ For Analog RGB and YPbPr inputs, Locked = 1 is returned only if the detected sensor format is that selected
- ▶ For HDMI input, Locked = 1 if the sensor is a valid HDMI source
- ▶ For composite (or S-Video) input, Locked = 1 if the sensor is a valid composite (or S-Video) source

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

3.6.9 Save Image

Saves the contents of the last image buffer as an image or raw data file, depending on filename.

Syntax

C/C++

```
int Hdv62_SaveImage (UINT Number, LPTSTR File-
Name)
```

C#

```
int SaveImage (uint Number, string FileName)
```

VB.Net

```
SaveImage (ByVal Number as UInteger, ByVal
FileName as String) As Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

FileName

The library supports:

- ▷ BMP: with extension *.bmp
- ▷ JPEG: with extension *.jpg or *.jpeg
- ▷ JIFF: with extension *.tif
- ▷ PNG: with extension *.png
- ▷ Raw data: other file type

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

3.6.10 Audio Stream

Acquires audio data pointer and size, normally called in callback after waiting for a frame event, or after calling One-Shot.

Syntax

C/C++

```
int Hdv62_GetAudioStream (UINT Number, AUDIO_STREAM_INFO& StreamInfo)
```

C#

```
int GetAudioStream (uint Number, out AUDIO_STREAM_INFO StreamInfo)
```

VB.Net

```
GetAudioStream (ByVal Number as UInteger, ByRef StreamInfo as AUDIO_STREAM_INFO) As Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

StreamInfo

Audio data structure and size as:

C/C++

```
typedef struct _ AUDIO_STREAM_INFO
{
   void *Data;
   UINT Size;
} AUDIO_STREAM_INFO;
```

C#

```
public struct AUDIO_STREAM_INFO
{
   public IntPtr Data;
   public uint Size;
}
```

VB.Net

```
Public Structure AUDIO_STREAM_INFO
   Public Data As IntPtr
   Public Size As UInteger
End Structure
```

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

3.7 Other Functions

3.7.1 Error Text

Retreives error text string

Syntax

C/C++

```
int Hdv62_GetErrorText (int code, char *Text)
```

C#

```
string GetErrorText (int code)
```

VB.Net

```
GetErrorText (ByVal code As Integer) As String
```

Parameter(s)

Code

Error code returned by other functions

Text

Error text string, for containment of which a buffer of maximum 160 bytes must be allocated

Return Value

C/C++

Always return 0

C#

Return the error text

VB.Net

Return the error text

3.8 EDID Functions

3.8.1 Ready Status

Sets or retrieves ready status of the EDID ROM.

Syntax

C/C++

```
int Hdv62_SetEdidReadyStatus (UINT Number,
UINT Status)
```

```
int Hdv62_GetEdidReadyStatus (UINT Number,
UINT& Status)
```

C#

```
int SetEdidReadyStatus (uint Number, uint Sta-
tus)
int GetEdidReadyStatus (uint Number, out uint
Status)
```

VB.Net

```
SetEdidReadyStatus (ByVal Number As UInteger,
ByVal Status As UInteger) As Integer
GetEdidReadyStatus (ByVal Number As UInteger,
ByRef Status As UInteger) As Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

Status

Indicates ready status of the EDID ROM, can be read by external device through DVI-I connector, with some external devices capable of resolution auto-adjustment accordingly. Content can be configured and EDID ROM set to ready state, based on:

0: EDID ROM is not ready

1: EDID ROM is ready

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

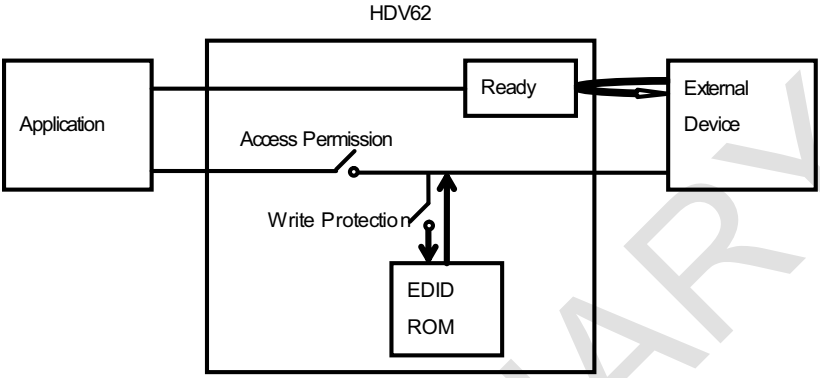


Figure 3-4: EDID ROM Architecture

3.8.2 Access Permission

Sets or retrieves application access to the EDID ROM.

Syntax

C/C++

```
int Hdv62_SetEdidAccessPermission (UINT Number, UINT Status)
int Hdv62_GetEdidAccessPermission (UINT Number, UINT& Status)
```

C#

```
int SetEdidAccessPermission (uint Number, uint Status)
int GetEdidAccessPermission (uint Number, out uint Status)
```

VB.Net

```
SetEdidAccessPermission (ByVal Number As UInteger, ByVal Status As UInteger) As Integer
GetEdidAccessPermission (ByVal Number As UInteger, ByRef Status As UInteger) As Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

Status

Indicates accessibility of the EDID ROM. Since access can be obtained by application or external device at the same time, to open EDID:

Establish access permission

Configure EDID ROM

Close access permission

Connect external device via DVI-I

From among the following values:

0: EDID ROM is not accessible

1: EDID ROM is accessible

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

3.8.3 Write Protection

Sets or retrieves writability of the EDID ROM.

Syntax

C/C++

```
int Hdv62_SetEdidWriteProtection (UINT Number,
UINT Status)
int Hdv62_GetEdidWriteProtection (UINT Number,
UINT& Status)
```

C#

```
int SetEdidWriteProtection (uint Number, uint
Status)
```

```
int GetEdidWriteProtection (uint Number, out
uint Status)
```

VB.Net

```
SetEdidWriteProtection (ByVal Number As UInte-
ger, ByVal Status As UInteger) As Integer
GetEdidWriteProtection (ByVal Number As UInte-
ger, ByRef Status As UInteger) As Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

Status

Indicates writeability of the EDID ROM. Any write protection must be cleared bfore data can be saved to the EDID. From among the following values:

0: EDID ROM is writeable

1: EDID ROM is protected

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

3.8.4 ROM Selector

Sets or retrieves the offset of the currently accessed EDID ROM.

Syntax

C/C++

```
int Hdv62_SetEdidRomSelector (UINT Number,
UINT Offset)
int Hdv62_GetEdidRomSelector (UINT Number,
UINT& Offset)
```

C#

```
int SetEdidRomSelector (uint Number, uint Offset)
int GetEdidRomSelector (uint Number, out uint Offset)
```

VB.Net

```
SetEdidRomSelector (ByVal Number As UInteger, ByVal Offset As UInteger) As Integer
GetEdidRomSelector (ByVal Number As UInteger, ByRef Offset As UInteger) As Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

Offset

Indicates the offset of the EDID ROM, with allowed values from 0 to 255.

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

3.8.5 ROM Read/Write

Sets or retrieves value of the EDID ROM.

Syntax

C/C++

```
int Hdv62_SetEdidRom (UINT Number, UINT Value)
int Hdv62_GetEdidRom (UINT Number, UINT& Value)
```

C#

```
int SetEdidRom (uint Number, uint Value)
int GetEdidRom (uint Number, out uint Value)
```

VB.Net

```
SetEdidRom (ByVal Number As UInteger, ByVal
Value As UInteger) As Integer

GetEdidRom (ByVal Number As UInteger, ByRef
Value As UInteger) As Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

Value

Indicates the value of the EDID ROM, with allowed values from 0 to 255.

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

3.9 Audio Format Control Functions

3.9.1 Audio Input

Sets or retrieves audio input configuration, only one of which is available at one time. Please see Section 1.4.2: Audio for connection details.

Syntax

C/C++

```
int Hdv62_SetAudioInput (UINT Number, UINT
Channel)

int Hdv62_GetAudioInput (UINT Number, UINT&
Channel)
```

C#

```
int SetAudioInput (uint Number, uint Channel)

int GetAudioInput (uint Number, out uint Chan-
nel)
```

VB.Net

```
SetAudioInput (ByVal Number As UInteger, ByVal
Channel As UInteger) As Integer

GetAudioInput (ByVal Number As UInteger, ByRef
Channel As UInteger) As Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

Channel

Audio source configuration, from among:

0: HDMI audio

1: SPDIF audio (Toslink)

2: SPDIF audio (RCA)

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

3.9.2 Audio Channels

Sets or retrieves the number of channels of the audio input.

Syntax

C/C++

```
int Hdv62_SetAudioChannels (UINT Number, UINT
Channels)

int Hdv62_GetAudioChannel (UINT Number, UINT&
Channels)
```

C#

```
int SetAudioChannels (uint Number, uint Chan-
nels)

int GetAudioChannels (uint Number, out uint
Channels)
```

VB.Net

```
SetAudioChannels (ByVal Number As UInteger,
ByVal Channels As UInteger) As Integer

GetAudioChannels (ByVal Number As UInteger,
ByRef Channels As UInteger) As Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

Channels

Specifies the number of channels of audio data, according to:

Stereo audio: 2

5.1 channel audio: 6

7.1 channel audio: 8

Currently SPDIF audio only supports up to 2 channels.

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

3.9.3 Samples Per Second

Sets or receives the sample frequency.

Syntax

C/C++

```
int Hdv62_SetAudioSamplesPerSec (UINT Number,
UINT Value)

int Hdv62_GetAudioSamplesPerSec (UINT Number,
UINT& Value)
```

C#

```
int SetAudioSamplesPerSec (uint Number, uint
Value)
```

```
int GetAudioSamplesPerSec (uint Number, out
uint Value)
```

VB.Net

```
SetAudioSamplesPerSec (ByVal Number As UInte-
ger, ByVal Value As UInteger) As Integer
GetAudioSamplesPerSec (ByVal Number As UInte-
ger, ByRef Value As UInteger) As Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

Value

Specifies sample frequency for each channel, such as 48000 when the sample frequency is 48kHz

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

3.9.4 Bits Per Sample

Sets or retrieves the number of bits per sample.

Syntax

C/C++

```
int Hdv62_SetAudioBitsPerSample (UINT Number,
UINT Value)
int Hdv62_GetAudioBitsPerSample (UINT Number,
UINT& Value)
```

C#

```
int SetAudioBitsPerSample (uint Number, uint
Value)
int GetAudioBitsPerSample (uint Number, out
uint Value)
```

VB.Net

```
SetAudioBitsPerSample (ByVal Number As UInteger, ByVal Value As UInteger) As Integer
GetAudioBitsPerSample (ByVal Number As UInteger, ByRef Value As UInteger) As Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

Value

Specifies the number of bits per sample, which the hardware extends to 24 bits if the audio source is 16 or 20 bit input.

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

3.9.5 Audio Capabilties

Sets or retrieves the supported audio properties of the card.

Syntax

C/C++

```
int Hdv62_GetAudioCapabilities (UINT Number, AUDIO_CAPABILITIES &Caps)
```

C#

```
int GetAudioCapabilities (uint Number, out AUDIO_CAPABILITIES Caps)
```

VB.Net

```
GetAudioCapabilities (ByVal Number As UInteger, ByRef Caps As AUDIO_CAPABILITIES) As Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

AUDIO_CAPABILITIES

Structure defined as:

C/C++

```
typedef struct _AUDIO_FORMATS
{
   unsigned long NumChannels;
   unsigned long *Channels;
   unsigned long NumBitsPerSample;
   unsigned long *BitsPerSamples;
   unsigned long NumSamplesPerSec;
   unsigned long *SamplesPerSecs;
} AUDIO_FORMATS;

typedef struct _AUDIO_CAPABILITIES
{
   AUDIO_FORMATS HdmiAudioFormats;
   AUDIO_FORMATS Spdif1AudioFormats;
   AUDIO_FORMATS Spdif2AudioFormats;
} AUDIO_CAPABILITIES;
```

C#

```
struct AUDIO_FORMATS
{
   public uint NumChannels;
   public IntPtr Channels;
   public uint NumBitsPerSample;
```

```
    public IntPtr BitsPerSamples;
    public uint NumSamplesPerSec;
    public IntPtr SamplesPerSecs;
}

struct AUDIO_CAPABILITIES
{
    public AUDIO_FORMATS HdmiAudioFormats;
    public AUDIO_FORMATS Spdif1AudioFormats;
    public AUDIO_FORMATS Spdif2AudioFormats;
}
```

VB.Net

```
Structure AUDIO_FORMATS
    Dim NumChannels As UInteger
    Dim Channels As IntPtr
    Dim NumBitsPerSample As UInteger
    Dim BitsPerSamples As IntPtr

    Dim NumSamplesPerSec As UInteger
    Dim SamplesPerSecs As IntPtr
End Structure
Public Structure AUDIO_CAPABILITIES
Dim HdmiAudioFormats As AUDIO_FORMATS
Dim Spdif1AudioFormats As AUDIO_FORMATS
Dim Spdif2AudioFormats As AUDIO_FORMATS
End Structure
```

Return Value

No error occurs if return value ≥ 0; if a negative value, please refer to Other Functions for return code error information.

4 DirectShow Programming Guide

.


NOTE:

Complete documentation for DirectShow application programming can be found at:
http://msdn.microsoft.com/en-us/library/dd390351(VS.85).aspx
by searching for “DirectShow”. If a DirectX SDK is installed, this documentation is also available from DirectX SDK Help. The most recent version of DirectShow SDK was moved to Windows SDK.

The goal when writing a DirectShow Application is to construct a filter graph by connecting several filters together, to execute tasks such as previewing or capturing video/audio, and multiplexing to write to a file. Each filter performs a single operation and pass data from its output is pinned to the input of the next filter in the graph.

To build a capture graph using a program, the interface pointer of the capture filter must first be obtained. The ADLINK HDV62A A/V Capture filter and the ADLINK HDV62A Crossbar filter can be obtained through the system device enumerator.

After holding an interface pointer to the capture filter object, method IGraphBuilder::AddSourceFilter adds the source filter object to the filter graph. IFilterGraph::AddFilter adds other downstream filters to the filter graph.

After filters are added, calling IFilterGraph::ConnectDirect or IGraphBuilder::Connect connect output pins from upstream filters to the input pins of the downstream filters.

Calling IAMCrossbar::Route to switch source channel, via methods IMediaControl::Run, IMediaControl::Pause or IMediaControl::Stop changes the filter state to running, paused or stopped.

Filters required for capturing video streams are

listed as follows, with detailed information for each and its pins. Example filter graphs for capturing video streams are also illustrated in this chapter with two ways of controlling the device driver.

4.1 Filters

This section lists the filters necessary to construct a filter graph for capturing a video stream.

4.1.1 Source Filters

ADLINK HDV62A A/V Capture

A WDM Streaming Capture Device, it is actually a kernel-mode KsProxy plug-in. An application can treat it simply as a filter. Use System Device Enumerator to add this filter to a filter graph.

Filter	ADLINK HDV62A A/V Capture
Filter CLSID	N/A
Filter Category Name	WDM Streaming Capture Devices
Filter Category	AM_KSCATEGORY_CAPTURE and/or Video Capture Sources
Capture Pin Supported Media Types	MEDIATYPE_Video Subtypes: MEDIASUBTYPE_RGB24 MEDIASUBTYPE_BGR30* MEDIASUBTYPE_RGB32 MEDIASUBTYPE_RGB8* MEDIASUBTYPE_YUV8* MEDIASUBTYPE_YUY2
Audio Capture Pin Supported Media Types	MEDIATYPE_AUDIO Subtypes: MEDIASUBTYPE_PCM
Exported Interfaces	IAMAnalogVideoDecoder IAMDroppedFrames IAMStreamConfig IAMVideoProcAmp IAdvance** IVideoFormat** ICardInfo** INotify

Filter	ADLINK HDV62A A/V Capture
Merit	MERIT_DO_NOT_USE

* Please see Section 4.3: Color Space

**Please see Section 4.4 Proprietary Interfaces

ADLINK HDV62A Crossbar Filter

If the device is a capture board, a crossbar filter is required to switch video sources. In hardware design, the crossbar can switch channel input of the same card.

Filter	ADLINK HDV62A Crossbar
Filter Category Name	WDM_Streaming Crossbar Devices
Exported Interfaces	IAMCrossbar
Input Pins	Pin Names: Video RGB In Video YRYBY In Video SerialDigital In Video Composite In Video SVideo In Audio SPDIF In Audio SPDIF In Audio SPDIF In
Output Pin	Pin Name: Video Decoder Out Audio Decoder Out


NOTE:

Video RGB In is an analog RGB signal from DVI-I connector

Video YRYBY In is an YPbPr signal from the external bracket connector

Video SerialDigital In is a HDMI signal from DVI-I connector

Video Composite In is a standard NTSC/PAL signal from external bracket

Video SVideo In is a standard NTSC/PAL signal from external bracket

1st Audio SPDIF In is a audio packet embedded into HDMI input

2nd Audio SPDIF In is a SPDIF audio signal from Toslink connector

3rd Audio SPDIF In is a SPDIF audio signal from RCA connector

Calling IAMCrossbar::Route switches the input channel

4.1.2 Example Graphs

The Microsoft DirectX SDK provides the GraphEdit debugging utility, which can simulate graph building. Desired filters can be chosen from the the Insert Filters command of the Graph menu. Filters are organized by categories. Insert Filter adds the filters to a graph, and two filters' pins can be selected by dragging from one

output pin to another. An arrow will be drawn if both pins agree on the connection.

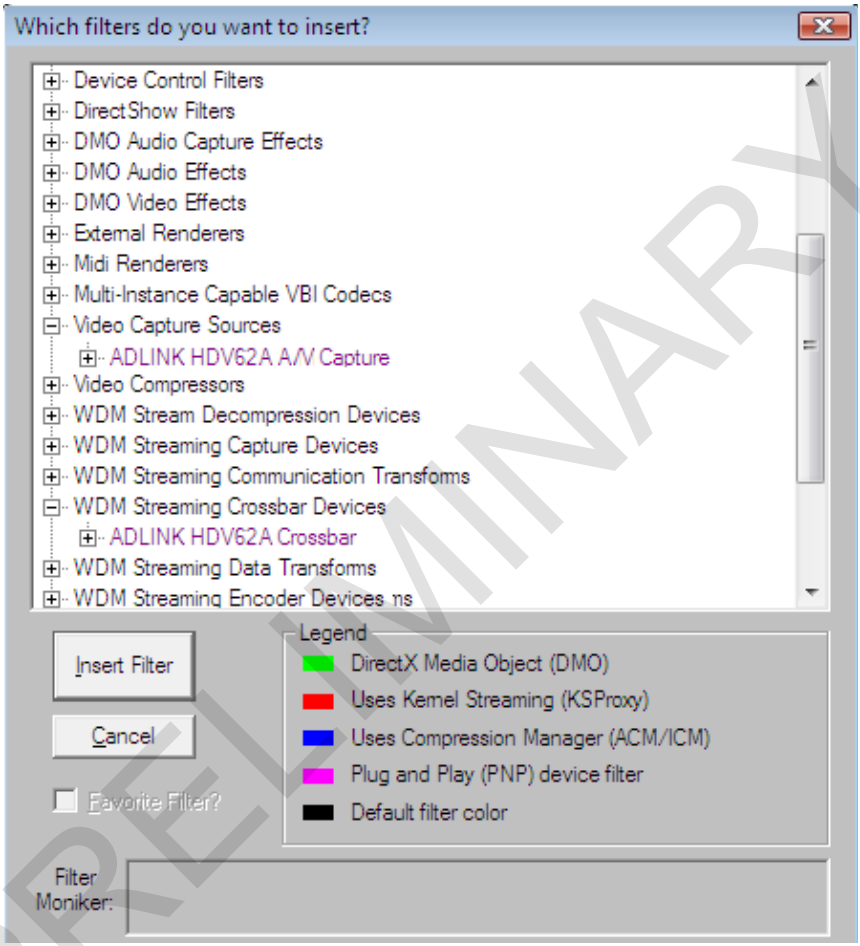


Figure 4-1: GraphEdit Insert Filters Dialog

After inserting the ADLINK HDV62A Video Capture filter and ADLINK HDV62A Crossbar filter, right clicking the rectangle and selecting Filter Properties allows the Property pages to be used to set video settings before connecting video pins to other filters, as shown.

HDV62A A/V Capture Filter

This selection generates the following series of Dialogs.

As shown, the System settings reflect the current configuration, with Sensor Format changed with different crossbar input. For supported sensor formats please see Sensor Format

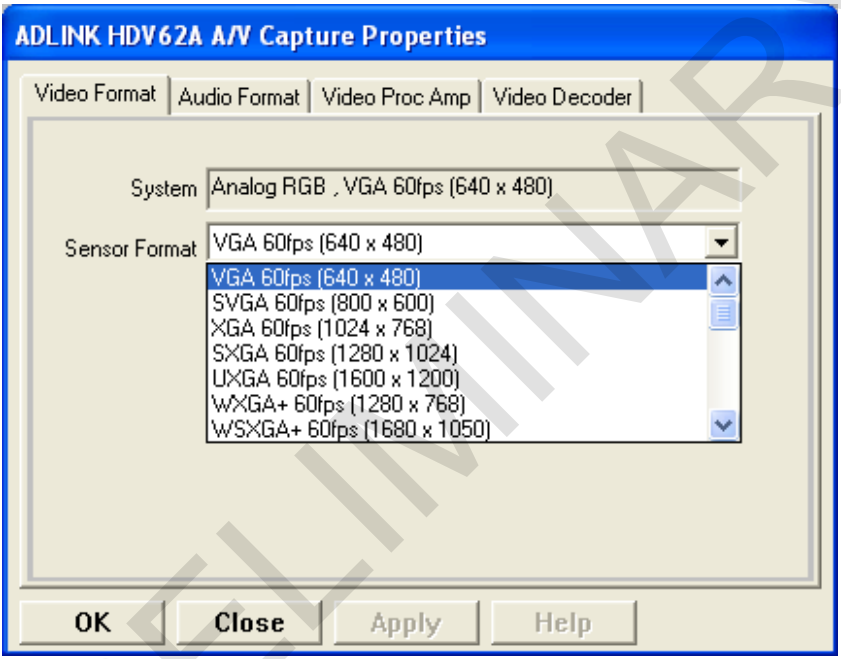


Figure 4-2: Video Format Dialog

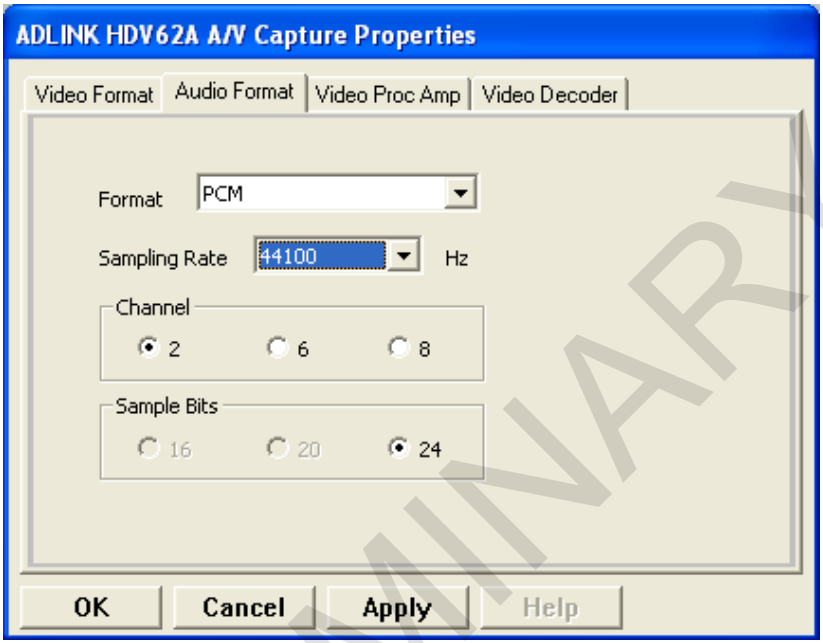


Figure 4-3: Audio Format Dialog

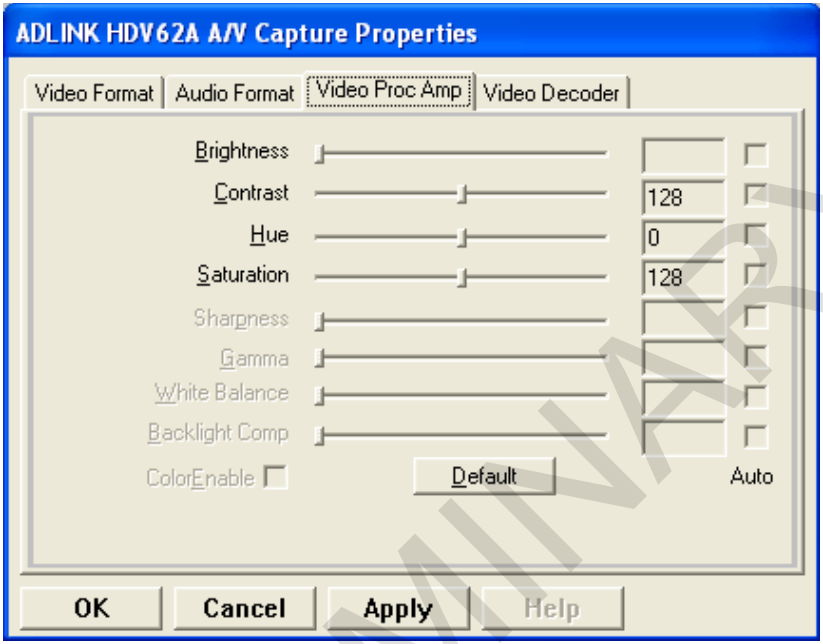


Figure 4-4: Video Proc Amp Dialog

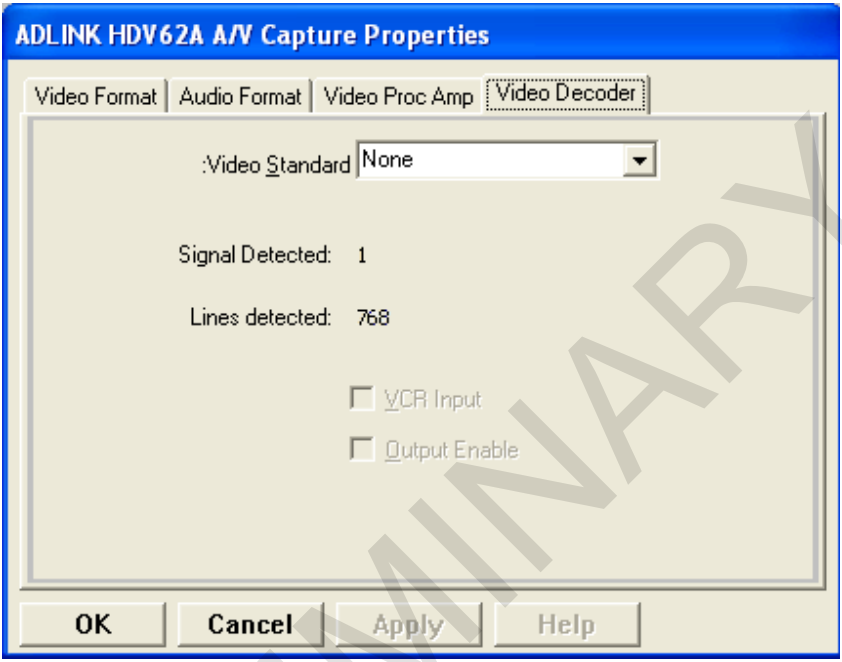


Figure 4-5: Video Decoder Dialog

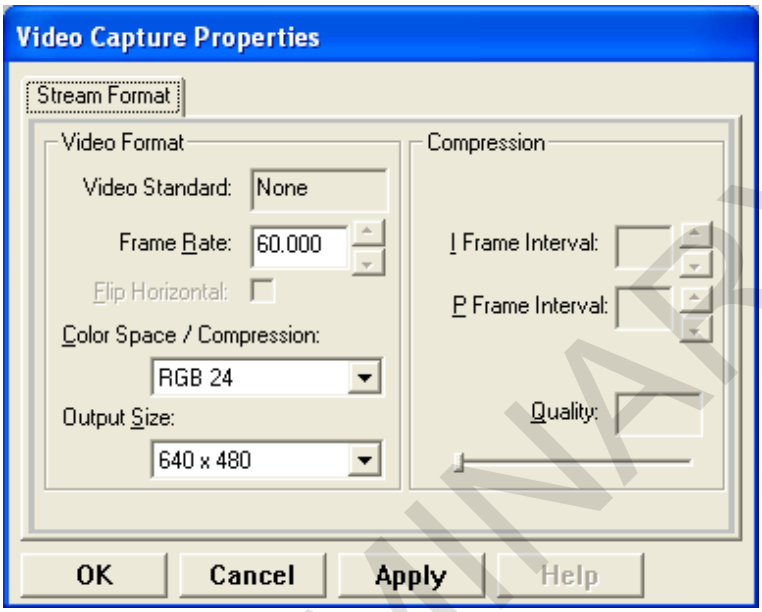


Figure 4-6: Capture Pin Properties Dialog

NOTE: In this dialog, a Color Space/Compression selection of BGRA indicates an actual setting of BGR30. For Supported Color Space parameters, please refer to Section 4.3 Color Space

ADLINK HDV62A Crossbar Filter

The relationship between inputs and outputs must be entered before the Capture pin of the ADLINK HDV62A A/V Capture filter is connected.

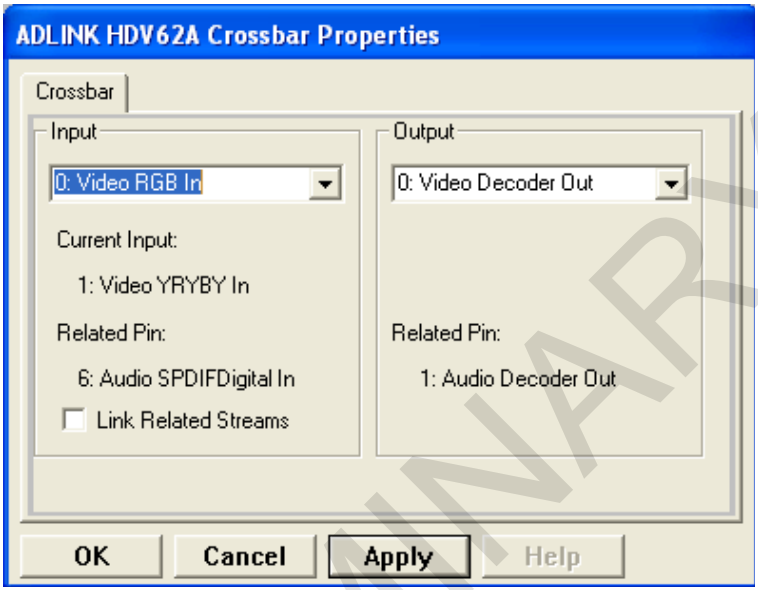


Figure 4-7: Crossbar Properties Dialog

Example Graph

To generate an example graph:

1. Open GraphEdit.exe.
2. Select 'Insert a filter into the graph' on the toolbar and enter 'ADLINK HDV62A A/V Video Capture', 'ADLINK HDV62A Crossbar', and 'Video Renderer' filters in the 'Video Capture Sources' group, the 'WDM Streaming

Crossbar Devices' group, and the 'DirectShow Filters' group, respectively.

3. Right click 'ADLINK HDV62A Crossbar' filter and select 'Filter Properties…', select the Input channel, and 'OK'.
4. Right click 'ADLINK HDV62A A/V Video Capture' filter and select 'Filter Properties…', select the Sensor Format, and 'OK'.
5. Right click Capture pin of 'ADLINK HDV62A A/V Video Capture' filter and select 'Pin Properties…', select Color Space/Compression, and 'OK'.
6. Drag 'Video Decoder Out' pin to 'Analog Video In' pin and from 'Capture' pin to 'Input' pin as shown.
7. Select 'Play the graph' from the toolbar to begin preview.

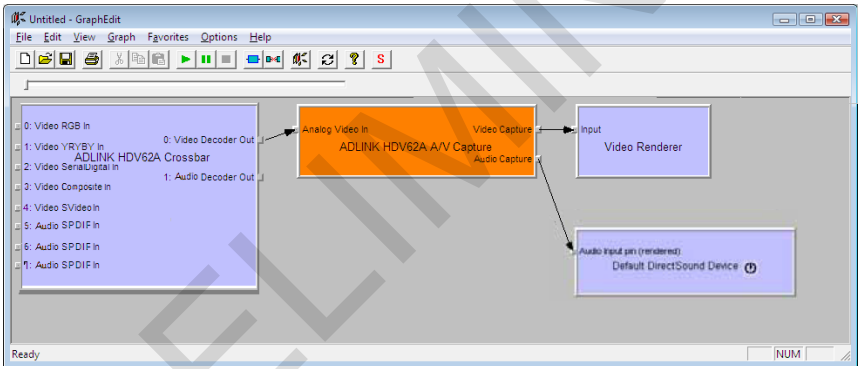


Figure 4-8: GraphEdit Interface

NOTE: If VMR substitutes for the default Video Renderer, the preview video shows a vertical mirror video, which can be resolved by inserting a Color Space Converter filter before.

4.2 Driver Control

The ADLINK HDV62A A/V Capture filter provides control of video configuration by either property pages or access to COM interfaces.

4.2.1 Property Pages

The driver provides two embedded property pages. To show these property pages, use Windows API: OleCreatePropertyFrame.

Details about Displaying a Filter's Property Page can be found on the Microsoft MSDN homepage.

Sample code for adding property pages is as follows.

```
// pFilter points to an ADLINK HDV62A Video Capture filter
// or an ADLINK HDV62A Crossbar filter

ISpecifyPropertyPages *pSpecify;
HRESULT hr;
hr = pFilter->QueryInterface(IID_ISpecifyPropertyPages, (void **)&pSpecify);
if (SUCCEEDED(hr))
{
FILTER_INFO FilterInfo;
pFilter->QueryFilterInfo(&FilterInfo);
FilterInfo.pGraph->Release();

CAUUID caGUID;
pSpecify->GetPages(&caGUID);
pSpecify->Release();

OleCreatePropertyFrame(
NULL,// Parent window
0,// x (Reserved)
0,// y (Reserved)
```

```
FilterInfo.achName,// Caption for the dialog box
1,// Number of filters
(IUnknown **)&m_pFilter,// Pointer to the filter
caGUID.cElems,// Number of property pages
caGUID.pElems,// Pointer to property page CLSIDs
0,// Locale identifier
0,// Reserved
NULL// Reserved
);
CoTaskMemFree(caGUID.pElems);
}
```

4.2.2 COM interfaces

The IAMVideoProcAmp interface of the standard DirectShow Interface provide acquisition or setting of incoming video signal attributes, such as brightness, contrast, saturation, and others. For other interfaces please refer to the DirectX SDK help.

```
// pFilter points to an ADLINK HDV62A A/V Video Capture
filter


IAMVideoProcAmp *pAmp;
HRESULT hr;
long contrast, flags;
hr    =    pFilter->QueryInterface(IID_IAMVideoProcAmp,
(void **)&pAmp);
if (SUCCEEDED(hr))
{
pAmp->Get(VideoProcAmp_Contrast, &contrast, &flags);

pAmp->Release();
}
```

ADLINK HDV62A Crossbar

The ADLINK HDV62A Crossbar filter, implementing an IAMCrossbar interface, routes signals from an analog or digital source to a video capture filter.

```
// pFilter points to an ADLINK HDV62A Crossbar filter


IAMCrossbar *pXbar;
HRESULT hr;
hr = pFilter->QueryInterface(IID_IAMCrossbar, (void **)&pXbar);
if (SUCCEEDED(hr))
{
// Route from input 1 to output 0
pXbar->Route(0, 1);
pXBar->Release();
}
```

4.3 Color Space

ADLINK HDV62A Video Capture supports the following color spaces:

('x' indicates an ignored bit)

MEDIASUBTYPE_RGB24 – 8bit R + 8bit G + 8bit B

DWORD	Pixel Data [31:0]			
	[31:24]	[23:16]	[15:8]	[7:0]
dw0	B1	R0	G0	B0
dw1	G2	B2	R1	G1
dw2	R3	G3	B3	R2

MEDIASUBTYPE_BGR30 – 10bit R + 10bit G + 10bit B

DWORD	Pixel Data [31:0]			
	[31:24]	[23:16]	[15:8]	[7:0]
dw0	xx+B[9:4]	B[3:0]+G[9:6]	G[5:0]+R[9:8]	R[7:0]


NOTE:

Compression (FOURCC code) is 'BGRA' and GUID is 41524742-0000-0010-8000-00AA00389B71

MEDIASUBTYPE_RGB32 – 8bit R + 8bit G + 8bit B + 8bit Alpha

DWORD	Pixel Data [31:0]			
	[31:24]	[23:16]	[15:8]	[7:0]
dw0	Alpha	R	G	B

MEDIASUBTYPE_RGB8 – 8bit Y (Grayscale)

DWORD	Pixel Data [31:0]			
	[31:24]	[23:16]	[15:8]	[7:0]
dw0	Y3	Y2	Y1	Y0

MEDIASUBTYPE_YUV8 – 8bit Y + 8bit Cb + 8bit Cr – YCbCr 4:4:4

DWORD	Pixel Data [31:0]			
	[31:24]	[23:16]	[15:8]	[7:0]
dw0	Y1	Cr0	Cb0	Y0
dw1	Cb2	Y2	Cr1	Cb1

DWORD	Pixel Data [31:0]			
	[31:24]	[23:16]	[15:8]	[7:0]
dw2	Cr3	Cb3	Y3	Cr2

Compression (FOURCC code) is 'YUV8' and GUID is 38565559-0000-0010-8000-00AA00389B71

MEDIASUBTYPE_YUY2 – 8bit Y + 8bit Cb/Cr – YCbCr 4:2:2

DWORD	Pixel Data [31:0]			
	[31:24]	[23:16]	[15:8]	[7:0]
dw0	Cr	Y	Cb	Y

YCbCr to RGB formula:

R = Y + 1.371(Cr-128)

G = Y – 0.698(Cr-128) – 0.336(Cb-128)

B = Y + 1.732(Cb-128)

4.4 Proprietary Interfaces

The following interfaces are specific to the ADLINK HDV62A A/V Video Capture filter, not being standard interfaces in DirectShow. COM interfaces, they can be acquired from the ADLINK HDV62A A/V Video Capture filter by calling IBaseFilter::QueryInterface.

4.4.1 IVideoFormat

Provides selection of sensor format, output format, horizontal delay, and video cropping.

Sensor Format

Read or write source format of the CCD sensor including video standard, resolution, and frame rate.

Syntax

C/C++

```
HRESULT WriteSensorFormat (UINT Format)
HRESULT ReadSensorFormat (UINT *Format)
```

C#

```
int WriteSensorFormat (uint Format)
int ReadSensorFormat (out uint Format)
```

VB.Net

```
WriteSensorFormat (ByVal Format As UInteger)
As Integer
ReadSensorFormat (ByRef Format As UInteger) As
Integer
```

Parameter(s)

Format

Format of the source sensor, with possible values for different crossbar input including:

Channel 0: Analog RGB from DVI-I connector

0: VGA 60 fps (640 x 480),

1: SVGA 60 fps (800 x 600),

2: XGA 60 fps (1024 x 768),

3: SXGA 60 fps (1280 x 1024),

4 : UXGA 60fps (1600 x 1200),

5 : WXGA+ 60fps (1280 x 768),

6 : WSXGA+ 60fps (1680 x 1050),

7 : XGA 75fps (1024 x 768),

8 : XGA 85fps (1024 x 768),

9 : SXGA 75fps (1280 x 1024),

10 : VGA 75fps (640 x 480),

11 : VGA 85fps (640 x 480),

12 : SVGA 75fps (800 x 600),

13 : SVGA 85fps (800 x 600),

14 : WXGA+ 75fps (1280 x 768),

15 : WXGA+ 60fps (1280 x 800),

16 : WXGA+ 75fps (1280 x 800),

17 : WXGA+ 85fps (1280 x 800),

18 : WXGA+ 85fps (1280 x 768),

19: SXGA 85 fps (1280 x 1024),

20 : XGA+ 75fps (1152 x 864),

21 : Wide XGA 60fps (1360 x768),

22 : SXGA+ 60fps (1400 x 1050),

23 : SXGA+ 75fps (1400 x 1050),

Channel 1: YPbPr from external bracket

0: 525i 30 fps (720 x 480 interlace, in frame per second),

1: 625i 25 fps (720 x 576 interlace, in frame per second),

2: 525p 60 fps (720 x 480 progressive),

3: 625p 50 fps (720 x 576 progressive),

5: 720p 50 fps (1280 x 720 progressive),

6: 720p 60 fps (1280 x 720 progressive),

7: 1080i 25 fps (1920 x 1080 interlace, in frame per second),

8: 1080i 30 fps (1920 x 1080 interlace, in frame per second),

9: 1080p 50 fps (1920 x 1080 progressive)

10: 1080p 60 fps (1920 x 1080 progressive)

Channel 2: HDMI from DVI-I connector

0: 720p 50 fps YCrCb In (1280 x 720 progressive),

1: 720p 60 fps YCrCb In (1280 x 720 progressive),

2: 1080i 25 fps YCrCb In (1920 x 1080 interlace, in frame per second),

3: 1080i 30 fps YCrCb In (1920 x 1080 interlace, in frame per second),

4: 1080p 25 fps YCrCb In (1920 x 1080 progressive)

5: 1080p 30 fps YCrCb In (1920 x 1080 progressive)

6: 1080p 50 fps YCrCb In (1920 x 1080 progressive)

7: 1080p 60 fps YCrCb In (1920 x 1080 progressive)

8: VGA 60 fps (640 x 480),

9: SVGA 60 fps (800 x 600),

10: XGA 60 fps (1024 x 768),

11: SXGA 60 fps (1280 x 1024),

12: UXGA 60 fps (1600 x 1200)

13: 720p 50 fps RGB In (1280 x 720 progressive),

14: 720p 60 fps RGB In (1280 x 720 progressive),

15: 1080i 25 fps RGB In (1920 x 1080 interlace, in frame per second),

16: 1080i 30 fps RGB In (1920 x 1080 interlace, in frame per second),

17: 1080p 25 fps RGB In (1920 x 1080 progressive)

18: 1080p 30 fps RGB In (1920 x 1080 progressive)

19: 1080p 50 fps RGB In (1920 x 1080 progressive)

20: 1080p 60 fps RGB In (1920 x 1080 progressive)

21: 1080p 24 fps YCrCb In (1920 x 1080 progressive)",

22: 1080p 48 fps YCrCb In (1920 x 1080 progressive)",

23: WXGA 60 fps YCrCb In (1360 x 768)",

24: 1200p 50 fps YCrCb In (1920 x 1200 progressive)",

25: 1200p 60 fps YCrCb In (1920 x 1200 progressive)",

26: 1080p 24 fps RGB In (1920 x 1080 progressive)",

27: 1080p 48 fps RGB In (1920 x 1080 progressive)",

28: WXGA 60 fps RGB In (1360 x 768)",

29: 1200p 50 fps RGB In (1920 x 120 progressive)",

30: 1200p 60 fps RGB In (1920 x 1200 progressive)",

31: 525i 30 fps YCrCb In (720 x 480 interlace, in frame per second)",

32: 625i 25 fps YCrCb In (720 x 576 interlace, in frame per second)",

33: 525i 30 fps RGB In (720 x 480 interlace, in frame per second)",

34: 625i 25 fps RGB In (720 x 576 interlace, in frame per second)",

35: WXGA+ 60 fps RGB In (1280 x 768)",

36: WSXGA+ 60 fps RGB In (1680 x 1050)",

37: VGA 60 fps YCrCb In (640 x 480)",

38: SVGA 60 fps YCrCb In (800 x 600)",

39: XGA 60 fps YCrCb In (1024 x 768)",

40: WXGA+ 60 fps YCrCb In (1280 x 768)",

41: WXGA 60 fps YCrCb In (1280 x 800)",

42: SXGA 60 fps YCrCb In (1280 x 1024)",

43: UXGA 60 fps YCrCb In (1600 x 1200)",

44: WSXGA+ 60 fps YCrCb In (1680 x 1050)"

Channel 3: Composite from external bracket

0: NTSC MJ

1: PAL 60

2: NTSC 4.43

3: PAL BGHID

4: PAL M

5: PAL Nc

Channel 4: S-Video from external bracket

0: NTSC MJ

1: PAL 60

2: NTSC 4.43

3: PAL BGHID

4: PAL M

5: PAL Nc


NOTE:

Final resolutions of channels 1, 2, and 3 can be acquired by calling ICardInfo::GetVideoCapabilities()

Return Value

No error occurs if return value ≥ 0; if a negative value, please call AMGetErrorText for return code error information.


NOTE:

Noise and black images lasting for a few seconds (actual time of noise and black images depends on the source) may occur if the input source is HDMI plus HDCP, and can be prevented by delaying commencing capture for a few seconds after initial connection of the HDMI device

Cropping

Reads or writes actual ouput resolution.

Syntax

C/C++

```
HRESULT WriteCropping (RECT Rt)
HRESULT ReadCropping (RECT * Rt)
```

C#

```
int WriteCropping (RECT Rt)
int ReadCropping (out RECT Rt)
```

VB.Net

```
WriteCropping (ByVal Rt As RECT) As Integer
ReadCropping (ByRef Rt As RECT) As Integer
```

Parameter(s)

Rt

A rectangle setting cropping the sensor image as shown in Figure 3-1. The rectangle must be located inside the sensor image, and actual image starts from [Rt.left, Rt.top] with width of [Rt.right – Rt.left] and height [Rt.bottom – Rt.top]. Actual image width must be a multiple of 16 pixels and actual height even. If sensor source is interlace, the top value in Rt must be even.

Return Value

No error occurs if return value ≥ 0; if a negative value, please call AMGetErrorText for return code error information

Horizontal Delay

Reads or writes horizontal delay of frame images, moving them left or right to remove black vertical lines, like X offset.

Syntax

C/C++

```
HRESULT WriteHDelay (INT Delay)
HRESULT ReadHDelay (INT * Delay)
```

C#

```
int WriteHDelay (int Delay)
int ReadHDelay (out int Delay)
```

VB.Net

```
WriteHDelay (ByVal Delay As Integer) As Inte-
ger
ReadHDelay (ByRef Delay As Integer) As Integer
```

Parameter(s)

Delay

Horizontal delay of frame images, with allowed values from -3 to 3. Each resolution has a default value that is called after channel and sensor format have been set.

Return Value

No error occurs if return value ≥ 0; if a negative value, please call AMGetErrorText for return code error information

AutoDetectSensorFormat

Acquires the format of the current selected video channel, reading back synchronization data from the source input. Aftert the channel is set to a supported input, after for least 3 frames (or 6 fields), the hardware can detect the signal.

Syntax

C/C++

```
HRESULT ReadAutoDetectSensorFormat(int *For-
mat)
```

C#

```
int ReadAutoDetectSensorFormat (out int For-
mat)
```

VB.Net

```
ReadAutoDetectSensorFormat (ByRef Format As
Integer) As Integer
```

Parameter(s)

Format

Read-back format of video sensor, the format is set as -1,as defined in Sensor Format if no sensor is detected or sensor format is not supported

Return Value

No error occurs if return value ≥ 0; if a negative value, please call AMGetErrorText for return code error information

Image Orientaton

Sets or retrieves orientation of images arranged in memory.

Syntax

C/C++

```
HRESULT WriteImageOrientation (UINT Value)
HRESULT ReadImageOrientation (UINT *Value)
```

C#

```
int WriteImageOrientation (uint Value)
int ReadImageOrientation (out uint Value)
```

VB.Net

```
WriteImageOrientation (ByVal Value As UInte-
ger) As Integer
ReadImageOrientation (ByRef Value As UInteger)
As Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

Value

Indicates the image orientation, from among:

0: Bottom-up, in which the image buffer starts with the bottom row of pixels, followed by the next row up, and so forth, with the top row of the image the last row in the buffer, such that the first byte in memory is the bottom-left pixel of the image. Physical layout of a bottom-up image is as shown.

E.g. Color space = RGB24

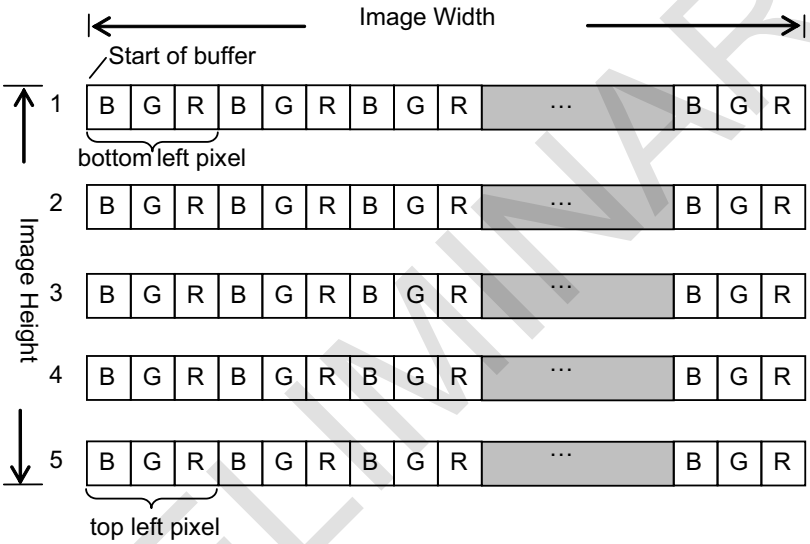


Figure 4-9: Bottom-up Image Orientation

1: Top-down, in which order of the row is reversed, with top row of the image the first row in memory, followed by the next row down, with bottom row of the image the last row in the buffer, such that first byte in memory is the top-left of the image. Physical layout of a top-down image is as shown.

E.g. Color space = RGB24

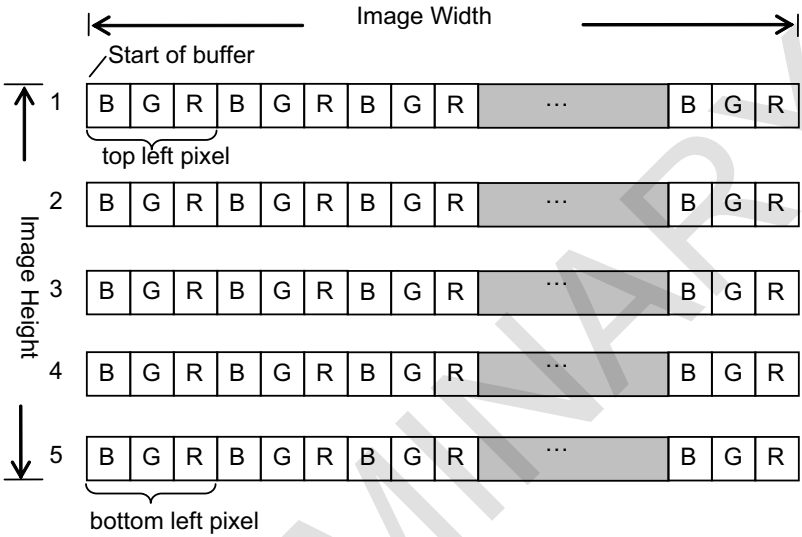


Figure 4-10: Top-down Image Orienation

Return Value

No error occurs if return value ≥ 0; if a negative value, please call AMGetErrorText for return code error information.

4.4.2 IAdvance

Provides access to EDID ROM.

EDID ROM Ready Status

Reads or writes Ready status of EDID ROM

Syntax

C/C++

```
HRESULT EDID_WriteReadyStatus (UINT Value)
HRESULT EDID_ReadReadyStatus (UINT *Value)
```

C#

```
int EDID_WriteReadyStatus (uint Value)
int EDID_ReadReadyStatus (out uint Value)
```

VB.Net

```
EDID_WriteReadyStatus (ByVal Value As UInteger) As Integer
EDID_ReadReadyStatus (ByRef Value As UInteger) As Integer
```

Parameter(s)

Value

Indicates whether or not the EDID ROM is ready, and is readable by external devices through DVI-I connector. Certain external devices can auto-adjust resolution by reading this EDID ROM. Content of the EDID ROM can be configured and set as Ready state, from among:

0: EDID ROM is not ready

1: EDID ROM is ready

Please see Figure 3-4: EDID ROM Architecture.

Return Value

No error occurs if return value ≥ 0; if a negative value, please call AMGetErrorText for return code error information

EDID ROM Access Permission

Reads or writes access to the EDID ROM

Syntax

C/C++

```
HRESULT    EDID_WriteAccessPermission    (UINT Value)
HRESULT     EDID_ReadAccessPermission     (UINT *Value)
```

C#

```
int EDID_WriteAccessPermission (uint Value)
int EDID_ReadAccessPermission (out uint Value)
```

VB.Net

```
EDID_WriteAccessPermission (ByVal Value As UInteger) As Integer
EDID_ReadAccessPermission (ByRef Value As UInteger) As Integer
```

Parameter(s)

Value

Indicates accessibility of the EDID ROM. The EDID ROM can be accessed by application or external device at the same time, and access permission must be opened, EDID ROM, configured, access permission closed, and the external device connected to the DVI-I connector. From among:

0: EDID ROM is not accessible

1: EDID ROM is accessible

Please see Section 3-4 EDID ROM Architecture.

Return Value

No error occurs if return value ≥ 0; if a negative value, please call AMGetErrorText for return code error information

Write Protection

Sets or retrieves writability of the EDID ROM.

Syntax

C/C++

```
HRESULT EDID_WriteWriteProtection (UINT Value)
HRESULT EDID_ReadWriteProtection (UINT *Value)
```

C#

```
int SetEdidWriteProtection (uint Number, uint Status)
```

```
int GetEdidWriteProtection (uint Number, out
uint Status)
```

VB.Net

```
SetEdidWriteProtection (ByVal Number As UInte-
ger, ByVal Status As UInteger) As Integer
GetEdidWriteProtection (ByVal Number As UInte-
ger, ByRef Status As UInteger) As Integer
```

Parameter(s)

Number

The number of the device, with allowed values from 0 to 15.

Status

Indicates writeability of the EDID ROM. Any write protection must be cleared bfore data can be saved to the EDID. From among the following values:

0: EDID ROM is writeable

1: EDID ROM is protected

Return Value

No error occurs if return value ≥ 0; if a negative value, please call AMGetErrorText for return code error information.

ROM Read/Write

Sets or retrieves value of the EDID ROM.

Syntax

C/C++

```
HRESULT EDID_WriteRom (UINT Offset, UINT
Value)
HRESULT EDID_ReadRom (UINT Offset, UINT
*Value)
```

C#

```
int EDID_WriteRom (uint Offset, uint Value)
```

```
int EDID_ReadRom (uint Offset, out uint Value)
```

VB.Net

```
EDID_WriteRom (ByVal Offset As UInteger , ByVal Value As UInteger) As Integer
EDID_ReadRom (ByVal Offset As UInteger , ByRef Value As UInteger) As Integer
```

Parameter(s)

Offset

Indicates the offset of the EDID ROM, with allowed values from 0 to 255.

Value

Indicates the value of the EDID ROM, with allowed values from 0 to 255.

Return Value

No error occurs if return value ≥ 0; if a negative value, please call AMGetErrorText for return code error information.

4.4.3 CardInfo

Version

Acquires miscellaneous versions

Syntax

C/C++

```
HRESULT GetHardwareVersion (UINT *Version)
HRESULT GetFirmwareVersion (UINT *Version)
HRESULT GetDriverVersion (UINT *Version)
```

C#

```
int GetHardwareVersion (out uint Version)
int GetFirmwareVersion (out uint Version)
int GetDriverVersion (out uint Version)
```

VB.Net

```
GetHardwareVersion (ByRef Version as UInteger)
As Integer
GetFirmwareVersion (ByRef Version as UInteger)
As Integer
GetDriverVersion (ByRef Version as UInteger)
As Integer
```

Parameter(s)

Version

Hexadecimal value in which each byte represents a version.

For driver version, Version = (Major << 24 + Minor << 16 + Revision << 8 + Release).

For example, if Version = 0x1000001, it represents “1.0.0.1”.

For firmware version, Version = (Year-2000) << 28 + Month << 24 + (Day / 16) << 20 + (Day % 16) << 16 + (Hour / 16) << 12 + (Hour % 16) << 8 + (Minute /16) << 4 + (Minute % 16).

To simplify, can be the output of printf in C language, as follows.

int Year = (Version >> 28) + 2000;

int Month = (Version >> 24) & 0x0F;

int Day = (Version >> 16) & 0xFF;

int Hour = (Version >> 8) & 0xFF;

int Minute = Version & 0xFF;

printf(“%d/%d/%x %x:%x”, Year, Month, Day, Hour, Minute);

For example, Version = 0x9b191407 represents “2009/11/19 14:07”.

For hardware version, the least 16 bits are carrier board version and the most 16 bits are daughter board version. If most 16 bits are all zero, the hardware is a single board.

Version of carrier board or daughter board = (Major << 8 + Minor).

For example, Version = 0xA1A2 represents carrier board version is “A2” and daughter board version is “A1”.

Return Value

No error occurs if return value ≥ 0; if a negative value, please call AMGetErrorText for return code error information

Card ID

Card ID as set by DIP switch.

Syntax

C/C++

```
HRESULT GetCardID (UINT* ID)
```

C#

```
int GetCardID (out uint ID)
```

VB.Net

```
GetCardID (ByRef ID As UInteger) As Integer
```

Parameter(s)

ID

Card ID is set by DIP switch on the HDV62A, with possible values from 0 to 15, allowing identification of individual cards when multiples are installed. For more information regarding setting Card ID, please see Section 1.6.1 Card ID Switch (SW3). Card ID can be left as default if individual assignment is undesired.

Return Value

No error occurs if return value ≥ 0; if a negative value, please call AMGetErrorText for return code error information

Video Capabilities

Acquires list of resolutions the card supports.

Syntax

C/C++

```
HRESULT                    GetVideoCapabilities
(RESOLUTION_CAPABILITIES *Caps)
```

C#

```
int                   GetVideoCapabilities(out
RESOLUTION_CAPABILITIES Caps)
```

VB.Net

```
GetVideoCapabilities     (ByRef     Caps     As
RESOLUTION_CAPABILITIES) As Integer
```

Parameter(s)

Supported resolutions defined as:

C/C++

```
typedef struct _SENSOR_RESOLUTION
{
   char Name[ 256 ];
   unsigned long Width;
   unsigned long Height;
   unsigned long FrameRate;
   unsigned long Interlace;
   unsigned long Hf;
   unsigned long HTotal;
   unsigned long Hsw;
   unsigned long Hbp;
   unsigned long Vf;
   unsigned long VTotal;
```

```
    unsigned long Vsw;
    unsigned long Vbp;
} SENSOR_RESOLUTION;

typedef struct _RESOLUTION_CAPABILITIES
{
    unsigned long NumRgbResolution;
    SENSOR_RESOLUTION *RgbResolutions;
    unsigned long NumYPbPrResolution;
    SENSOR_RESOLUTION *YPbPrResolutions;
    unsigned long NumHdmiResolution;
    SENSOR_RESOLUTION *HdmiResolutions;
} RESOLUTION_CAPABILITIES;
```

C#

```
struct SENSOR_RESOLUTION
{
    // name
        [MarshalAs(UnmanagedType.ByValTStr,  SizeConst  =
256)]
    public string Name;

    // video setting
    public uint Width;
    public uint Height;
    public uint FrameRate;
    public uint Interlace;

    // video timing
```

```
    public uint Hf;// horizontal frequency
    public uint HTotal;// horizontal total line
    public uint Hsw;// horizontal sync width
    public uint Hbp;// horizontal back porch
    public uint Vf;// vertical frequency
    public uint VTotal;// vertical total line
    public uint Vsw;// vertical sync width
    public uint Vbp;// vertical back porch
}

struct RESOLUTION_CAPABILITIES
{
    public uint NumRgbResolution;
    public IntPtr RgbResolutions;

    public uint NumYPbPrResolution;
    public IntPtr YPbPrResolutions;

    public uint NumHdmiResolution;
    public IntPtr HdmiResolutions;
}
```

VB.Net

```
Structure SENSOR_RESOLUTION

<MarshalAs(UnmanagedType.ByValTStr,
SizeConst:=256)> _
    Dim Name As String
```

```
    Dim Width As UInteger
    Dim Height As UInteger
    Dim FrameRate As UInteger
    Dim Interlace As UInteger

    Dim Hf As UInteger      ' horizontal frequency
    Dim HTotal As UInteger  ' horizontal total line
    Dim Hsw As UInteger     ' horizontal sync width
    Dim Hbp As UInteger     ' horizontal back porch
    Dim Vf As UInteger      ' vertical frequency
    Dim VTotal As UInteger  ' vertical total line
    Dim Vsw As UInteger     ' vertical sync width
    Dim Vbp As UInteger     ' vertical back porch

End Structure

Structure RESOLUTION_CAPABILITIES

    Dim NumRgbResolution As UInteger
    Dim RgbResolutions As IntPtr

    Dim NumYPbPrResolution As UInteger
    Dim YPbPrResolutions As IntPtr

    Dim NumHdmiResolution As UInteger
    Dim HdmiResolutions As IntPtr

End Structure
```

A variable of structure RESOLUTION_CAPABILITIES must be set and the elements of RgbResolutions, YPbPrResolutions, and HdmiResolutions set as null pointers. Calling GetVideoCapabilities() acquires sizes of NumRgbResolution, NumYPbPrResolution, and NumHdmiResolution.

Sufficient buffers must then be allocated to contain all resolutions of SENSOR_RESOLUTION array, and GetVideoCapabilities() called again to get all resolution arrays.

Return Value

No error occurs if return value ≥ 0; if a negative value, please call AMGetErrorText for return code error information.

Audio Capabilties

Sets or retrieves the supported audio properties of the card.

Syntax

C/C++

```
HRESULT GetAudioCapabilities
(AUDIO_CAPABILITIES *Caps)
```

C#

```
int         GetAudioCapabilities          (out
AUDIO_CAPABILITIES Caps)
```

VB.Net

```
GetAudioCapabilities     (ByRef     Caps     As
AUDIO_CAPABILITIES) As Integer
```

Parameter(s)

AUDIO_CAPABILITIES

Structure defined as:

C/C++

typedef struct _AUDIO_FORMATS

{

```
    unsigned long NumChannels;
    unsigned long *Channels;
    unsigned long NumBitsPerSample;
    unsigned long *BitsPerSamples;
    unsigned long NumSamplesPerSec;
    unsigned long *SamplesPerSecs;
} AUDIO_FORMATS;

typedef struct _AUDIO_CAPABILITIES
{
    AUDIO_FORMATS HdmiAudioFormats;
    AUDIO_FORMATS Spdif1AudioFormats;
    AUDIO_FORMATS Spdif2AudioFormats;
} AUDIO_CAPABILITIES;
```

C#

```
struct AUDIO_FORMATS
{
    public uint NumChannels;
    public IntPtr Channels;
    public uint NumBitsPerSample;
    public IntPtr BitsPerSamples;
    public uint NumSamplesPerSec;
    public IntPtr SamplesPerSecs;
}

struct AUDIO_CAPABILITIES
{
    public AUDIO_FORMATS HdmiAudioFormats;
```

```
    public AUDIO_FORMATS Spdif1AudioFormats;
    public AUDIO_FORMATS Spdif2AudioFormats;
}
```

VB.Net

```
Structure AUDIO_FORMATS
    Dim NumChannels As UInteger
    Dim Channels As IntPtr
    Dim NumBitsPerSample As UInteger
    Dim BitsPerSamples As IntPtr

    Dim NumSamplesPerSec As UInteger
    Dim SamplesPerSecs As IntPtr
End Structure
Public Structure AUDIO_CAPABILITIES
Dim HdmiAudioFormats As AUDIO_FORMATS
Dim Spdif1AudioFormats As AUDIO_FORMATS
Dim Spdif2AudioFormats As AUDIO_FORMATS
End Structure
```

A variable of structure AUDIO_CAPABILITIES must be declared and elements of Channels, BitsPerSamples, and SamplesPerSecs set as null pointers. Calling GetAudioCapabilities() acquires the sizes of NumChannels, NumBitsPerSample, and NumSamplesPerSec, sufficient buffers to contain all settings of the array must be allocated, and then GetAudioCapabilities() called again to get all arrays.

4.5 Build Environment Settings

4.5.1 Include Files

Applications must include the files as shown.

Include File	Description
DShow.h	The header file is required for all C++ applications.
Hdv62Proxy.h	The header file is required for all C++ applications.
Hdv62Guids.h	The header file is required for all C++ applications
DirectShowLib	Imports this name space for all Microsoft .Net application.
Hdv62ProxyLib	Imports this name space for all Microsoft .Net application.

4.5.2 Library Files

Applications must include the library files as shown.

Library File	Description
Strmiids.lib	Exports class identifiers (CLSIDs) and interface identifiers (IIDs). All C++ applications require this library.
Quartz.lib	Exports the AMGetErrorText function for C++ applications. If you do not call this function, this library is not required.
DirectShowLib-2005.dll	The class library of DirectShow is required for all Microsoft .Net applications.
Hdv62ProxyLib.dll	The class library of the interfaces of HDV62A is required for all Microsoft .Net applications.

The libraries for Microsoft .Net applications work and test on .Net Framwork 2.0, Microsoft Visual Studio 2005 is recco-mended for building .Net applications.

4.5.3 Microsoft Visual C++

For VC++, the build environment must be set up prior to building, as follows.

1. Open the solution file (baseclasses.sln) or the project file (baseclasses.dsw) under %DXSDK%\Samples\C++\DirectShow\BaseClasses and build it.
2. Add the paths to the include directory inproject settings:
   %DXSDK%\include
   %DXSDK%\Samples\C++\DirectShow\BaseClasses
3. Add the paths to the additional library directory in project settings:
   %DXSDK%\Lib
   %DXSDK%\Samples\C++\DirectShow\BaseClasses\Release
   %DXSDK%\Samples\C++\DirectShow\BaseClasses\Debug

For the above, %DXSDK% is the installation path of DirectX SDK.

4.5.4 .Net Programming Users

Microsoft DirectShow provides only C++ programming. .Net users must convert DirectShow COM objects to .net classes. Source codes and samples from a supporting sourceforge project can be downloaded from http://sourceforge.net/projects/directshownet/. Samples dedicated to HDV62A cards are also provided in the installation directory.

Important Safety Instructions

For user safety, please read and follow all **instructions**, **WARNINGS**, **CAUTIONS,** and **NOTES** marked in this manual and on the associated equipment before handling/operating the equipment.

- Read these safety instructions carefully.
- Keep this user's manual for future reference.
- Read the specifications section of this manual for detailed information on the operating environment of this equipment.
- When installing/mounting or uninstalling/removing equipment:
  - Turn off power and unplug any power cords/cables.
- To avoid electrical shock and/or damage to equipment:
  - Keep equipment away from water or liquid sources;
  - Keep equipment away from high heat or high humidity;
  - Keep equipment properly ventilated (do not block or cover ventilation openings);
  - Make sure to use recommended voltage and power source settings;
  - Always install and operate equipment near an easily accessible electrical socket-outlet;
  - Secure the power cord (do not place any object on/over the power cord);
  - Only install/attach and operate equipment on stable surfaces and/or recommended mountings; and,
  - If the equipment will not be used for long periods of time, turn off and unplug the equipment from its power source.

- Never attempt to fix the equipment. Equipment should only be serviced by qualified personnel.

A Lithium-type battery may be provided for uninterrupted, backup or emergency power.

Risk of explosion if battery is replaced with one of an incorrect type. Dispose of used batteries appropriately.

- Equipment must be serviced by authorized technicians when:
  - The power cord or plug is damaged;
  - Liquid has penetrated the equipment;
  - It has been exposed to high humidity/moisture;
  - It is not functioning or does not function according to the user's manual;
  - It has been dropped and/or damaged; and/or,
  - It has an obvious sign of breakage.

Getting Service

Contact us should you require any service or assistance.

ADLINK Technology, Inc.
Address: 9F, No.166 Jian Yi Road, Zhonghe District
New Taipei City 235, Taiwan
新北市中和區建一路 166 號 9 樓
Tel: +886-2-8226-5877
Fax: +886-2-8226-5717
Email: service@adlinktech.com

Ampro ADLINK Technology, Inc.
Address: 5215 Hellyer Avenue, #110, San Jose, CA 95138, USA
Tel: +1-408-360-0200
Toll Free: +1-800-966-5200 (USA only)
Fax: +1-408-360-0222
Email: info@adlinktech.com

ADLINK Technology (China) Co., Ltd.
Address: 上海市浦东新区张江高科技园区芳春路 300 号 (201203)
300 Fang Chun Rd., Zhangjiang Hi-Tech Park,
Pudong New Area, Shanghai, 201203 China
Tel: +86-21-5132-8988
Fax: +86-21-5132-3588
Email: market@adlinktech.com

ADLINK Technology Beijing
Address: 北京市海淀区上地东路 1 号盈创动力大厦 E 座 801 室(100085)
Rm. 801, Power Creative E, No. 1,
Shang Di East Rd., Beijing, 100085 China
Tel: +86-10-5885-8666
Fax: +86-10-5885-8626
Email: market@adlinktech.com

ADLINK Technology Shenzhen
Address: 深圳市南山区科技园南区高新南七道 数字技术园
A1 栋 2 楼 C 区 (518057)
2F, C Block, Bldg. A1, Cyber-Tech Zone, Gao Xin Ave. Sec. 7,
High-Tech Industrial Park S., Shenzhen, 518054 China
Tel: +86-755-2643-4858
Fax: +86-755-2664-6353
Email: market@adlinktech.com

LiPPERT ADLINK Technology GmbH
Address: Hans-Thoma-Strasse 11, D-68163, Mannheim, Germany
Tel: +49-621-43214-0
Fax: +49-621 43214-30
Email: emea@adlinktech.com

ADLINK Technology, Inc. (French Liaison Office)
Address: 15 rue Emile Baudot, 91300 Massy CEDEX, France
Tel: +33 (0) 1 60 12 35 66
Fax: +33 (0) 1 60 12 35 66
Email: france@adlinktech.com

ADLINK Technology Japan Corporation
Address: 〒101-0045 東京都千代田区神田鍛冶町 3-7-4
神田 374 ビル 4F
KANDA374 Bldg. 4F, 3-7-4 Kanda Kajicho,
Chiyoda-ku, Tokyo 101-0045, Japan
Tel: +81-3-4455-3722
Fax: +81-3-5209-6013
Email: japan@adlinktech.com

ADLINK Technology, Inc. (Korean Liaison Office)
Address: 서울시 서초구 서초동 1675-12 모인터빌딩 8 층
8F Mointer B/D,1675-12, Seocho-Dong, Seocho-Gu,
Seoul 137-070, Korea
Tel: +82-2-2057-0565
Fax: +82-2-2057-0563
Email: korea@adlinktech.com

ADLINK Technology Singapore Pte. Ltd.
Address: 84 Genting Lane #07-02A, Cityneon Design Centre,
Singapore 349584
Tel: +65-6844-2261
Fax: +65-6844-2263
Email: singapore@adlinktech.com

ADLINK Technology Singapore Pte. Ltd. (Indian Liaison Office)
Address: 1st Floor, #50-56 (Between 16th/17th Cross) Margosa Plaza,
Margosa Main Road, Malleswaram, Bangalore-560055, India
Tel: +91-80-65605817, +91-80-42246107
Fax: +91-80-23464606
Email: india@adlinktech.com

ADLINK Technology, Inc. (Israeli Liaison Office)
Address: 6 Hasadna St., Kfar Saba 44424, Israel
Tel: +972-9-7446541
Fax: +972-9-7446542
Email: israel@adlinktech.com