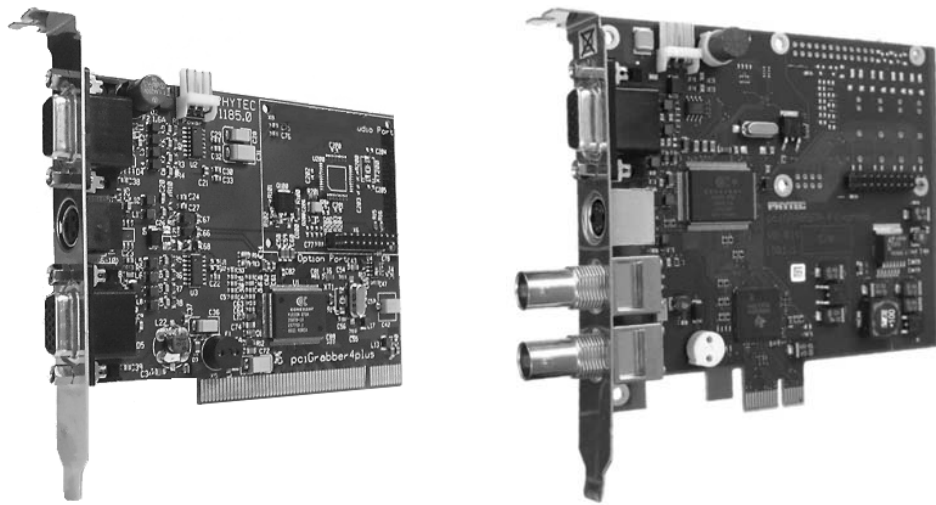


PCI LOCAL BUS Grabber-4plus
PCI LOCAL BUS Grabber-4express



Hardware-Manual

Edition: December 2009

In this manual are descriptions for copyrighted products that are not explicitly indicated as such. The absence of the trademark (™) and copyright (©) symbols does not imply that a product is not protected. Additionally, registered patents and trademarks are similarly not expressly indicated in this manual.

The information in this document has been carefully checked and is believed to be entirely reliable. However, PHYTEC Messtechnik GmbH assumes no responsibility for any inaccuracies. PHYTEC Messtechnik GmbH neither gives any guarantee nor accepts any liability whatsoever for consequential damages resulting from the use of this manual or its associated product. PHYTEC Messtechnik GmbH reserves the right to alter the information contained herein without prior notification and accepts no responsibility for any damages which might result.

Additionally, PHYTEC Messtechnik GmbH offers no guarantee nor accepts any liability for damages arising from the improper usage or improper installation of the hardware or software. PHYTEC Messtechnik GmbH further reserves the right to alter the layout and/or design of the hardware without prior notification and accepts no liability for doing so.

© Copyright 2009 PHYTEC Messtechnik GmbH, D-55129 Mainz.

Rights - including those of translation, reprint, broadcast, photomechanical or similar reproduction and storage or processing in computer systems, in whole or in part - are reserved. No reproduction may occur without the express written consent from PHYTEC Messtechnik GmbH.

	EUROPE	NORTH AMERICA
Address:	PHYTEC Technologie Holding AG Robert-Koch-Str. 39 D-55129 Mainz GERMANY	PHYTEC America LLC 203 Parfitt Way SW, Suite G100 Bainbridge Island, WA 98110 USA
Ordering Information:	+49 (800) 0749832 order@phytec.de	1 (800) 278-9913 sales@phytec.com
Technical Support:	+49 (6131) 9221-31 support@phytec.de	1 (800) 278-9913 support@phytec.com
Fax:	+49 (6131) 9221-33	1 (206) 780-9135
Web Site:	http://www.phytec.de	http://www.phytec.com

Edition No. 6

1	Introduction.....	7
1.1	General information on the manual's structure	7
1.2	Quickstart	8
Part 1 Installation and Start-Up		
2	Fields of Application and Safety Instructions	10
2.1	Notes on CE-Conformance and Immunity against Interference	11
3	pciGrabber-4<i>plus</i>	13
3.1	Scope of Delivery (pciGrabber-4 <i>plus</i>)	13
3.2	Accessories	13
3.3	Technical Data VD-009(-X1).....	15
3.4	Addresses and Resources	19
3.5	Socket Pinout.....	20
3.5.1	Composite Inputs	20
3.5.2	S-Video Connection	23
3.5.3	Power Supply Output	24
3.5.4	I/O Pin	25
3.5.5	RS6 variant.....	27
3.5.6	Option Port	28
3.5.7	I ² C Interface	29
3.6	Installing and Starting Up the Framegrabber Card	30
3.6.1	Installing the Framegrabber Card.....	30
3.7	Connecting Video Sources	32
3.7.1	Video Connections	34
3.7.2	The Video/Power Cable	37
3.7.3	The S-Video Cable	38
3.7.4	The Composite Connectors	38
4	pciGrabber-4<i>express</i>	39
4.1	Scope of delivery (pciGrabber-4 <i>express</i>).....	39
4.2	Accessories	39
4.3	Technical Data VD-011	41
4.4	Addresses and Resources	45
4.5	Socket Pinout.....	46
4.5.1	Composite Inputs	46
4.5.2	S-Video Connection	49
4.5.3	Power Supply Output	50
4.5.4	I/O Pin	51
4.5.5	RS6 variant.....	53
4.5.6	Option Port	54
4.5.7	I ² C Interface	55

4.6	Installing and Starting Up the Framegrabber Card.....	56
4.6.1	Installing the Framegrabber Card	56
4.7	Connecting Video Sources.....	58
4.7.1	Video Connections.....	60
4.7.2	The Video/Power Cable.....	63
4.7.3	The S-Video Cable.....	64
4.7.4	The Composite Connectors.....	64
5	Installing the Driver.....	65
5.1	Additional Drivers (optional).....	67
6	Start-Up the Framegrabber with the Demo Application.....	68
6.1	Installing the Demo Application.....	68
6.2	Description of the Demo Software	69
6.3	Demo Software - Detailed Description.....	74
6.4	Image Control	80
6.5	Additional Functions of the <i>Image</i> Dialog.....	81
6.6	Crosshair function (Overlay)	82
6.7	Basic Parameters	82
6.8	Special Functions	84
6.9	Storing Images, Ending the Program.....	92
6.10	Getting Started with Linux.....	92

Part2 Programm's Guide

7	Driver Software.....	95
7.1	Technical Basics	96
7.1.1	Block Diagram of the pciGrabber-4plus.....	96
7.1.2	The Video Signal and Capturing Process	99
7.1.3	Transfer and storage of color	101
7.1.4	Data storage by DMA and RISC-Program	103
7.2	Driver for Microsoft Windows	107
7.2.1	Requirements	108
7.2.2	Installation of the VxD-Driver (Windows '95)	108
7.2.3	Application of the Device Driver for Windows NT4.0	112
7.2.4	Application of the Device Driver for Windows' 98 TM and Windows' 2000 / XP / VISTA.....	116
7.2.5	Application of the DLL.....	117
7.2.6	Application of the Windows 95/98/ME/XP/VISTA TM Windows NT4.0 TM / Windows 2000 TM DLLs	118
7.2.7	Programming under Delphi	119
7.2.8	Description of the DLL's Functions.....	121

7.3	Driver for DOS Applications	178
7.3.1	Premises	178
7.3.2	Development Platform	179
7.3.3	Functions of the DOS Driver <i>PCI4GRAB</i>	180
7.3.4	Program Example DOS	200
8	Changes to the pciGrapper-4 and Compatilby	204
8.1	Changes between the pciGrabber-4 and pciGrabber-4 <i>plus</i>	204
8.2	Changes between the pciGrabber-4 <i>plus</i> and pciGrabber-4 <i>express</i>	209
9	Trouble-Shooting	210
	Index.....	216

Index of Figures

Figure 1:	Accessory Cables pciGrabber-4plus	14
Figure 2:	Connectors of the pciGrabber-4plus	20
Figure 3:	Standard Connections for the I/O Pin as Input	25
Figure 4:	Standard Connections for the I/O Pin as Output.....	26
Figure 5:	Pin Formation of the Option Port.....	28
Figure 6:	Inserting the Card into the PCI Slot.....	31
Figure 7:	Set up of the power supply feature.....	32
Figure 8:	Overview of the pciGrabber-4plus Connectors	33
Figure 9:	Video Connector Cables - (Description and PHYTEC part number)	34
Figure 10:	Connectors for the VD-009 Model	35
Figure 11:	Connectors for the VD-009-X1 Model	36
Figure 12:	Connecting a Camera (VCAM 110-x) to the Video Power Cable (example)	37
Figure 13:	Accessory Cables pciGrabber-4express.....	40
Figure 14:	Connectors of the pciGrabber-4express.....	46
Figure 15:	Standard Connections for the I/O Pin as Input	51
Figure 16:	Standard Connections for the I/O Pin as Output.....	52
Figure 17:	Pin Formation of the Option Port.....	54
Figure 18:	Inserting the Card into the PCI Express- Slot.....	57

Figure 19: Set up of the power supply feature	58
Figure 20: Overview of the pciGrabber-4express Connectors.....	59
Figure 21: Video Connector Cables - (Description and PHYTEC part number).....	60
Figure 22: Connectors for the VD-011 Model (part 1).....	61
Figure 23: Connectors for the VD-011 Model (part 2).....	62
Figure 24: Connecting a Camera (VCAM 110-x) to the Video Power Cable (An Example)	63
Figure 25: PHYTEC Installation Menu.....	68
Figure 26: Overview of the Demo Application.....	69
Figure 27: Menu Option: Image.....	70
Figure 28: Configuring the Image Parameters	71
Figure 29: Live Image from the Video Source.....	72
Figure 30: „Image Setting“ Menu	74
Figure 31: Creating a Full Image: Two Fields, Each with 7 rows	77
Figure 32: Comb Effect That Occurs with Quick Moving Objects	78
Figure 33: The Image Control Window	80
Figure 34: Basic Settings Menu	82
Figure 35: Histogram.....	84
Figure 36: Color Meter.....	85
Figure 37: Arithmetics Menu	87
Figure 38: Selecting the Normalization Factor	88
Figure 39: Number of Images	88
Figure 40: I/O Test Menu.....	89
Figure 41: Option Port Menü	90
Figure 42: DIP switches Menu.....	91
Figure 43: Relays Menu	91
Figure 44: Block diagram.....	96
Figure 45: Block diagram.....	98
Figure 46: Interlaced image (Example with 9 Lines).....	99

Figure 47: Fields and Frames	100
Figure 48: Moving Objects Cause Comb Effects	100
Figure 49: Pixel- and Control Data Flow (Overview).....	105
Figure 50: Folders for Window's Driver	107
Figure 51: Windows'95 Registry Editor	109
Figure 52: Adding of a VxD-Entry.....	110
Figure 53: Setting Up the VxD.....	111
Figure 54: Windows NT Registration Editor	113
Figure 55: Entering a Device Driver	113
Figure 56: Configuring the Driver.....	114
Figure 57: Scaling and Cropping.....	144
Figure 58: Example of Scaling: Only the ppl value is changed	145
Figure 59: Color Format of the pciGrabber-4plus/express.....	152
Figure 60: Return Values of Data_Present'	157
Figure 61: Timing Diagram of the Return Parameters of Data_Present().....	158

Index of Tables

Table 1 :	Pin Assignment of the HD-DB-15 Sockets, Model VD-009 ...	21
Table 2:	Pin Assignments of the HD-DB-15 Sockets,	
	Model VD-009-X1	22
Table 3	Connection of the S-Video Input to the HD-DB-15 Socket	23
Table 4:	Connection of the I/O Pin to the Combi Socket	25
Table 5:	Relays / multi-pin connector X14	27
Table 6:	Pin Assignment for the Option Port.....	28
Table 7	Connecting the I ² C Interface to the Combi Socket	29
Table 8:	Pin Assignments of the pciGrabber-4express (VD-011)	47
Table 9	Connection of the S-Video Input to the HD-DB-15 Socket	49
Table 10:	Connection of the I/O Pin to the Combi Socket	51
Table 11:	Relays / multi-pin connector X14	53

Table 12:	Pin Assignment for the Option Port.....	54
Table 13	Connecting the PC Interface to the Combi Socket.....	55
Table 14:	Required Memory Space of One Pixel for the Different Modi	153
Table 15:	Memory Requirements for a Pixel in the Single Mode	189
Table 16:	Pin Assignment for the HD-DB-15 Sockets,..... Model VD-009	206
Table 17	Pin Assignments for the HD-DB-15 Sockets, Model VD-009-X1	207
Table 18:	Pin Assignment of the Option Port - Connectors (Both Models)	207

1 Introduction

Thank you for purchasing the pciGrabber from PHYTEC Messtechnik GmbH. This manual explains how to install the PC-Card and gives further information on the software.

The following table shows an overview of the types and models which are described in this manual.

The main differences between the pciGrabbers listed are the bus system used and the number of video inputs.

TYPE	Article-No.:	Bus-System	Inputs
pciGrabber-4 <i>plus</i>	VD-009	PCI	9 x Comp. 1 x S-Video
pciGrabber-4 <i>plus</i>	VD-009-RS6	PCI	9 x Comp. 1 x S-Video
pciGrabber-4 <i>plus</i>	VD-009-X1	PCI	3 x Comp. 1 x S-Video
pciGrabber-4 <i>plus</i>	VD-009-X1-RS6	PCI	3 x Comp. 1 x S-Video
pciGrabber-4 <i>express</i>	VD-011	x1 PCI Express	3 x Comp. 1 x S-Video
pciGrabber-4 <i>express</i>	VD-011-RS6	x1 PCI Express	3 x Comp. 1 x S-Video

Models with the suffix –RS6 provide four additional relays and a DIP-switch. Details can be found in chapter 3.5.5 and chapter 4.5.5.

1.1 General information on the manual's structure

This manual applies to various models. Below is a brief description, which chapter refers to which model.

pciGrabber-4*plus*:

Chapter 1, 2, 3, 5, 6, 7 and 8

pciGrabber-4*express*:

Chapter 1, 2, 4, 5, 6, 7 and 8

As you can see, chapters 1, 2, 5, 6, 7 and 8 are for both models, while chapter 3 gives information exclusively on the pciGrabber-4*plus* and chapter 4 on the pciGrabber-4*express*.

1.2 Quickstart

The following explains which chapter should be read if you are interested in one of the specific topics listed below.

General information about the hardware:

pciGrabber-4plus:

- Chapter 3

pciGrabber-4express:

- Chapter **Fehler! Verweisquelle konnte nicht gefunden werden.**

Hardware installation:

pciGrabber-4plus:

- Chapter 3.6

pciGrabber-4express:

- Chapter **Fehler! Verweisquelle konnte nicht gefunden werden.**

Driver installation:

pciGrabber-4plus und *pciGrabber-4express*:

- Chapter **Fehler! Verweisquelle konnte nicht gefunden werden.**

Installing and using the Demo Software:

pciGrabber-4plus und *pciGrabber-4express*:

- Chapter 6

Programming:

pciGrabber-4plus und *pciGrabber-4express*:

- Chapter 7

Part 1

Installation and Start-Up

2 Fields of Application and Safety Instructions

Please take care to comply with the specified operating conditions when using the *pciGrabber-4plus/express*. Read these instructions carefully before start-up.

- The *pciGrabber-4plus/express* is designed to digitize video signals from standard TV-cameras. Signals from composite-video cameras which comply with the CCIR B, G, H, I standard and the sub standard CCIR B, G, H, I/PAL can be processed. In addition signals compliant to CCIR M/NTSC can be applied. The camera signals are also applicable according to the S-video standard with separate luma and chroma signals.
The images are digitized in real time. The image data are transferred via the PCI- / PCI Express bus. The transfer rate corresponds to the access time specified for the PCI master mode of the PC.
The effective transfer rate must be sufficient to handle the volume of the image data, otherwise information might be lost.
- The *pciGrabber-4plus/express* is determined for the utilization with a standard PC, meaning an office computer with a usual housing. The *pciGrabber-4plus* has to be plugged into a PCI-slot with busmaster capability and the *pciGrabber-4express* accordingly into a PCI Express-slot. The framegrabber must have a reliable connection with the housing and grounding (PE).
The board is designed to operate in dry and dustless environment. For applications in industrial environment you have to consider to take additional protective arrangements especially against radio interference and safety hazards.
- The application of the framegrabber board in safety areas, for aerospace and for nuclear or military purposes requires our examinations and our agreement.
- For industrial applications all rules for prevention of accidents and the rules of the employer's liability insurance association for electrical facilities are to observe.

- Before starting the operation of the framegrabber, it must be ensured, that the device is appropriate for the application and the specific location. In case of doubt, you should ask experts or the manufacturer.
- The product has to be protected from hard shocks and vibrations. Eventually the device has to be padded or cushioned, but the ventilation may not be obstructed.
- In need of repair only a specialist is to be asked, who uses the original spare parts. For the installation of the grabber, use only tested and approved cables. Only radio shielded cables should be utilized.

2.1 Notes on CE-Conformance and Immunity against Interference

Upon delivery, the *pciGrabber-4plus/express* meets all CE-specifications for household, office, manufacturing and industry. Any modifications of the framegrabber without permission of the manufacturer will result in the cancellation of the CE-certificate.

CE-conforming use of the framegrabber is only maintained by utilizing CE-certified cables. These cables can be separately purchased from PHYTEC as accessories for the *pciGrabber-4plus/express* (see section 3.2 and 4.2). If other cables are installed the user must ensure CE-conformity.

If the user plans to connect the *pciGrabber-4plus/express* with other cables, it is recommended that these cables are fitted with an anti-interference clamp or comparable interference suppression devices. The clamp should be placed about 5 cm from the framegrabber and, the cable should be looped twice through the clamp.

For video cables a ferrite type # 742.711.4 from Würth, Kupferzell, Germany is suitable.

The cable shielding has to be connected to the connector shell to obtain an optimum of shielding.

The *pciGrabber-4plus/express* was tested for a standard PC environment. If the device should be used in a different environment, it has to be examined if additional radio shielding is necessary.

Caution:

Please pay attention, that significant interference peaks (ESD) to the video signal or video ground might damage the input of the *pciGrabber-4plus/express*.

In areas with high interference level, for example in industrial areas and using long feed lines, additional precautions have to be taken to suppress interference. Long video cables, or mounting the components for image processing into plants and machines, can cause the exposition to balancing currents, which have to be eliminated from the input of the *pciGrabber-4plus/express* by appropriate arrangements. PHYTEC does not assume any liability for damages that occur due to incorrect connections of the signal source.

3 pciGrabber-4plus

3.1 Scope of Delivery (pciGrabber-4plus)

- a PCI-card
- Installation CD with
 - Demo software (Windows‘ 95/98/ME/XP, NT4.0, 2000 and Windows VISTA)
 - Driver library for DOS (with DOS4GW)
 - Driver software for Windows‘ 95/98/ME/XP, NT4.0, 2000 and Windows VISTA
 - Twain driver for applications with Twain interfaces
- this pciGrabber-4plus/ pciGrabber-4express manual

3.2 Accessories

The following pciGrabber-4plus accessories may be ordered from PHYTEC:

- Composite connector cable for five cameras (upper socket of VD-009) – **not compatible with VD009-X1** – HD-DB15 to 5 x BNC-plug, length approx. 2 m – order number WK012
- Composite connector cable for four cameras and a power supply output (12 VDC) for a camera (lower socket VD-009, or upper socket VD-011) HD-DB15 to 4 x BNC-plug and 1 x power plug, length approx. 2 m – Order number WK022
- S-Video connector cable for connection of color cameras with a 4-pin Mini-DIN plug (S-Video output). Length, approx. 2 m – Order number: WK051
- Combi connector cable for color cameras with S-Video connection and 12 V power supply. HD-DB15 to 1 x Mini DIN plug and 1 x power supply (open ends) Compatible with the VCAM 110, 120. Length approx. 2 m. – Order number WK075.

- Replacement fuse 1.6A T TR5 for camera power supply (receptacle F2) – Order number KF012
- Replacement fuse 500mA T TR5 for camera power supply (receptacle F1) – Order number KF014



Figure 1: Accessory Cables pciGrabber-4plus

3.3 Technical Data VD-009(-X1)

Physical

Dimensions:	120 x 80 x 20 mm plus backplate and slot
Data Bus:	PCI bus 5 V, Master slot required (PCI Rev. 2.1 compliant)
Power Supply:	+5 V (250 mA idle, 300 mA digitizing) -12 V (40 mA, not with model VD-009-X1) (from PCI bus)
Inputs:	<u>Model VD-009:</u> 9 composite video inputs, 75 Ω , 1 V _{ss} ¹ 1 S-Video input 75 Ω (0.7 V _{ss} / 0.3 V _{ss}) <u>Model VD-009-X1:</u> 3 composite video inputs, 75 Ω , 1 V _{ss} ¹ 1 S-Video input 75 Ω (0.7 V _{ss} / 0.3 V _{ss})
Video Format:	PAL (B,G,H,I), NTSC (M) or corresponding CCIR monochrome format
Synchronization:	Composite sync. or sync to Y-signal external synchronization not featured
Data Format:	16 Mio. colors: RGB32, RGB24, YcrCb 4:2:2, YcrCb 4:1:1 64,000 colors: RGB16 32,000 colors: RGB15 256 gray shades: Y8 gray scale

¹: If an S-Video input is not being used, an extra composite input is available

Image

Resolution: maximum 768 x 576 pixels (PAL)
or 640 x 480 pixels (NTSC)
Resolution is freely scalable in X and Y directions
up to 14:1

Image Transfer

Rate: Field transfer 20 ms (either odd or even fields)
Full frame transfer 40 ms
Image transfer to the main memory in real time
(bus master transfer)

Used

Resources: 4 kByte main memory (register field)
/INTA

Image control: Gamma correction (selectable)
Brightness (+/- 50 %)
Contrast (0 % ... 235 %)
Color saturation (U: 0...201 %, V: 0...283 %)
Hue (+/- 90°, only with NTSC)

Image Storage: 630 Byte FIFO on-board,
Real time storage in the PC main memory
Even-/odd field memory separated or
Common full frame memory (selectable)

Ports: 12-bit parallel I/O, TTL signal (multi-purpose)

Parameter	Symbol	Min	Max
Input High Voltage	V_{IH}	2,0 V	5 V
Input Low Voltage	V_{IL}	-0,5 V	0,8 V
Output High Voltage	V_{OH}	2,4 V	-
Output Low Voltage	V_{OL}	-	0,4 V
Input Low Current	I_{IL}	-	-70 μ A
Input High Current	I_{IH}	-	70 μ A

1 I/O Port (transistor-buffered, 28 V/0.8 A_{max})

Parameter	Symbol	Min	Max
Input High Voltage	V_{IH}	2,0 V	28 V
Input Low Voltage	V_{IL}	-0,5 V	0,5 V
Output High Voltage	V_{OH}	5 V	25 V
Output Low Voltage	V_{OL}	0 V	1,4 V
Input Low Current	I_{IL}	-	-700 μ A
Input High Current	I_{IH}	-	70 μ A
Output HiZ Current	I_{OZ}	-	500 μ A
Output On Current	I_{OON}	-	800 mA
Switching frequency	f_{IO}		200 Hz

1 I²C interface (Master)

Parameter	Symbol	Min	Max
Transmission rate ¹	f_{I2C}	99,2 kHz	396,8 kHz
Input High Voltage	V_{IH}	3,5 V	5 V
Input Low Voltage	V_{IL}	-0,5 V	1,5 V
Hysteresis	V_{hys}	0,2 V	
Input High Current	I_{IH}	-	10 μ A
Input Low Current	I_{IL}	-	-10 μ A
Output Low Voltage	V_{OL}	-	0,4 V

Relays: (version -RS6 only)
4 Relay-outputs (N.O., 24V, 1A max.)

DIP-Switch: (version -RS6 only)
quadruple DIP-switch

¹: frequencies can be selected software

Connectors:

Model VD-009

HD-DB-15 socket 1:	5 composite video inputs
HD-DB-15 socket 2:	4 composite video inputs 1 S-Video chroma input 1 I ² C interface 1 I/O connection, driven 1 output for camera power supply, +12 V/1.5 A _{max}
Mini DIN socket:	S-Video input
Pin header row 2x10: (<i>not on the backplate</i>)	GPIO port, 12 x TTL I/O I ² C interface I/O connection, driven
Pin header row 2 x 4: (<i>not on the backplate</i>)	4 Relay-outputs (optional only VD-009-RS6)

Model VD-009-X1

HD-DB-15 socket 1:	not connected
HD-DB-15 Socket 2:	4 Composite video inputs 1 S-Video chroma input 1 I ² C interface 1 I/O connection, driven 1 output for camera power supply, +12 V/1.5 A _{max}
Mini DIN socket:	S-Video input
Pin header row 2 x 10: (<i>not on backplate</i>)	GPIO port, 12 x TTL I/O I ² C interface I/O connection, driven
Pin header row 2 x 4: (<i>not on backplate</i>)	4 Relay-outputs (optional only VD-009-X1-RS6)

3.4 Addresses and Resources

The pciGrabber-4plus occupies a range of 4 kBytes in the main memory of the PC for the local registers. The address range is automatically specified by the BIOS and no hardware setup (jumper settings) is required.

Several pciGrabber-4plus can be installed in one system. The boards are configured automatically by the BIOS for different addresses.

It is not possible to determine the specific address for a certain board. The base address of each board can be obtained by the PCI-BIOS. For the pciGrabber-4plus the driver software determines the address via the BIOS and defines a device number. The driver also can determine the number of boards within the system and is able to control each board by its particular device number.

It is not possible to determine which board will be specified by which device number. This will be determined by the PCI-BIOS and the architecture of the PC-motherboard. Usually the addresses are allocated in sequence of the numbering of the PCI slots. This might deviate for different manufacturers. To solve that problem the RS6 version can be used. It allows to give every framegrabber an explicit address by different DIP-switch settings.

The pciGrabber-4plus will activate an interrupt in case of certain events or a distinct operational status.

The framegrabber is a *single function device* so only the interrupt line /INTA of the PCI-bus can be used. To this PCI-bus-interrupt an interrupt of the PC is allocated via the BIOS, so that the program can react to this event.

The source of the interrupt can be determined from the interrupt status register of the framegrabber.

Several boards can trigger the same interrupt /INTA, it must be determined which board caused the interrupt.

3.5 Socket Pinout

Note:

The following description of the framegrabber's connectors is intended as a technical reference.

3.5.1 Composite Inputs

All composite video sources with an output level of $1 V_{SS}$ and an impedance of 75Ω can be used. For more information on supported video standards, please refer to *section 3.3*.

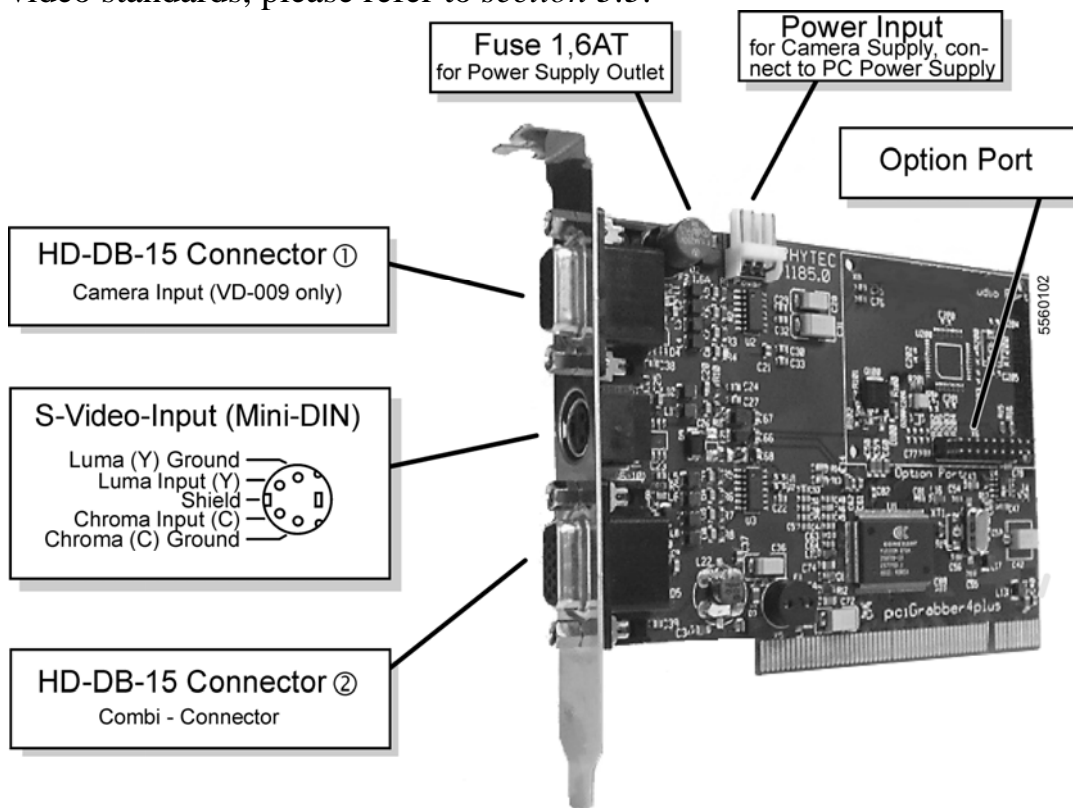


Figure 2: Connectors of the pciGrabber-4plus

- **Version VD-009**

The composite video signal is connected to the framegrabber via a multiplexer. This connection yields nine inputs, providing access to two HD-DB-15 sockets (①,②) (refer to *Figure 2*). Table 1 depicts the pin assignment.

- **Version VD-009-X1**

Three composite inputs available to the framegrabber are located on the *lower* HD-DB-15 socket ②.

The input assignment for the channel numbers is as follows:

pciGrabber-4plus with 9 Composite Inputs (VD-009)

HD-DB-15 ① (X1)		HD-DB-15 ② (X2)	
Pin	Function	Pin	Function
1	Composite Input 1	1	Composite Input 6
2	Composite Input 2	2	Composite Input 7
3	Composite Input 3	3	Composite Input 8
4	S-Video: Luma	4	S-Video: Chroma
5	Signal Ground	5	Signal Ground
6	Signal Ground	6	Signal Ground
7	Signal Ground	7	Signal Ground
8	Signal Ground	8	Signal Ground
9		9	I/O-Pin
10	Signal Ground	10	Pwr Supply Ground(-)
11	Signal Ground	11	Signal Ground
12	I ² C Bus: SDA	12	I ² C Bus: SDA
13	Composite Input 4	13	Composite Input 9
14	Composite Input 5	14	+12 V out (Camera supply)
15	I ² C Bus: SCL	15	I ² C Bus: SCL

Table 1 : Pin Assignment of the HD-DB-15 Sockets, Model VD-009

pciGrabber-4plus with 3 Composite Inputs (VD-009-X1)

HD-DB-15 ① (X1)		HD-DB-15 ② (X2)	
Pin	Function	Pin	Function
1		1	Composite Input 1
2		2	Composite Input 2
3		3	S-Video: Luma
4	S-Video: Luma	4	S-Video: Chroma
5	Signal Ground	5	Signal Ground
6	Signal Ground	6	Signal Ground
7	Signal Ground	7	Signal Ground
8	Signal Ground	8	Signal Ground
9		9	I/O-Pin
10	Signal Ground	10	Pwr Supply Ground(-)
11	Signal Ground	11	Signal Ground
12	I ² C Bus: SDA	12	I ² C Bus: SDA
13		13	Composite Input 3
14		14	+12 V out (Camera supply)
15	I ² C Bus: SCL	15	I ² C Bus: SCL

Table 2: Pin Assignments of the HD-DB-15 Sockets, Model VD-009-X1

In addition to the composite video inputs, a power output is available. The power output enables a +12 V power source from the host PC to be connected to a camera. Refer to section 3.5.3, „Power Supply Output“.

PHYTEC offers connecting cables that enable the application of the video signal via BNC plugs. The upper connector ① (video inputs 1-5) fits the cable WK012, and the lower connector ② (video inputs 6-9 and power supply) fits the cable WK022 (see section 3.2).

Caution:

Exchanging the cables can result in a connection of the power output, +12 V, to a camera's video output. This might destroy the camera or the framegrabber.

If the power output is not intended for use, then remove fuse F1 or F2, in order to avoid a power supply of +12 V at connector ②.

3.5.2 S-Video Connection

The advantage of this design is the separate conduct of brightness and color signal. This prevents disturbing Moiré effects for fine image structures and improves the resolution of the color image.

There are two options to connect a S-Video source to the pciGrabber-4plus:

- Direct an S-Video signal to the 4-pin Mini DIN socket (X3). The socket is wired corresponding to the S-Video standard (*refer to Figure 2*). The connection of the camera is made by a standard S-Video cable.
- Using a special cable (i.e. WK075), it is possible to connect an S-Video camera to the HD-DB-15 socket ②. Connecting the S-Video camera in this manner allows additional power supplies of +12 V, or additional signals, to be directed by the same connection cable. If such a cable is being used, then the following pins are to be connected (connection of the power supply is optional):

HD-DB-15 ② (X2)	
Pin	Function
4	S-Video: Chroma
5	Signal Ground
6	Signal Ground
10	Pwr Supply Ground(-)
13	S-Video: Luma
14	+12 V out (Camera supply)

Table 3 Connection of the S-Video Input to the HD-DB-15 Socket

Caution:

Both S-Video inputs can **not** be connected at the same time. Either the Mini DIN input or the HD-DB-15 socket ② can be used. The user must select in the application software which socket is connected to the S-Video source. (Automatic selection of the socket is also possible).

3.5.3 Power Supply Output

The pciGrabber-4plus provides a power supply output of +12 V at pin 14 on the lower HD-DB-15 socket, ② for the connected camera(s). Therefore, a supplemental power supply is not necessary if the camera is installed in the vicinity of the PC. The maximum current load is 1.5 A

Note:

In order to use the power supply output, the pciGrabber-4plus must be connected to the PC power supply. Connect a free power supply cable from the power supply to the connector plug (X7), located on the framegrabber. 3,5“ floppy drive power supply connectors are suitable for use.

The +12V voltage supplied to the plug X7 is provided at the HD-DB-15 plug ② from the framegrabber.

For more information on installing the power supply cables, *refer to section 3.6.1.*

A miniature fuse, F2, protects the output. Additional replacement fuses can be ordered from PHYTEC (part number KF012).

Regarding the output current, please adhere to the output power supply (+12V) specifications of the PC power supply.

3.5.4 I/O Pin

A universal I/O pin is located at pin 9 of the HD-DB-15 plug ②. This I/O pin is universal in the sense that it can be used as either input or output. In order to use this I/O pin, ensure that the application software supports this function.

Using the I/O pin as input enables the application software to receive control signals. The input can be polled from the program and is not connected to any special functions.

When using the function as output, the application software is able to convey control signals to other devices. The output can be set to Hi-Z state or set to ground connection by the software.

I/O Pin Functions

- **Input**

An external voltage (with reference to Ground) can be connected to the I/O pin. If the voltage is between 0 V and 0.5 V, then the program reads a „0“. If the voltage is between 2 V and 28 V, then the logic signal level is set to „1“. The positive connection must be at terminal 9 (see *Table 4*).

HD-DB-15 ② (X2)	
Pin	Function
9	I/O Pin (+)
10	Ground (-)

Table 4: Connection of the I/O Pin to the Combi Socket

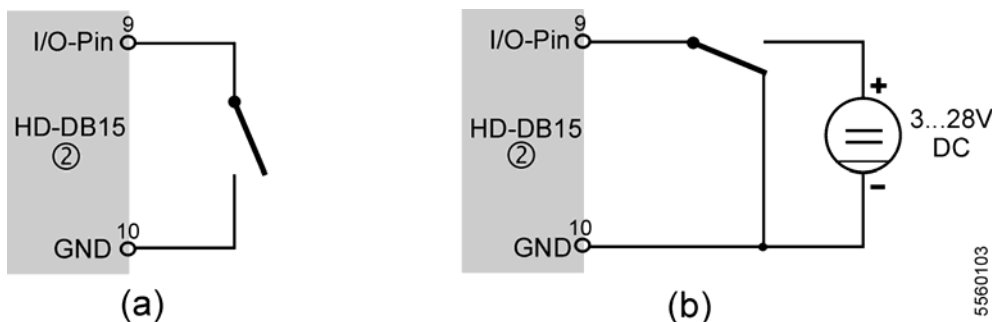


Figure 3: Standard Connections for the I/O Pin as Input

- **Output**

If the pin is used as an output, it has the following behavior:

If the software sets the pin to logic „1“ output, then pin 9 is connected to Ground (i.e. pin 10), via a transistor.

If the software sets the pin to logic „0“, then the transistor is in high impedance (Hi-Z) - state and thus, there is no connection between pin 9 and Ground.

In order to use the output, an external power supply, in the range of 5 V and 28 V, is required. It is also possible to use the power supply pin (pin 14) for power supply. *Figure 4* depicts two possible connections.

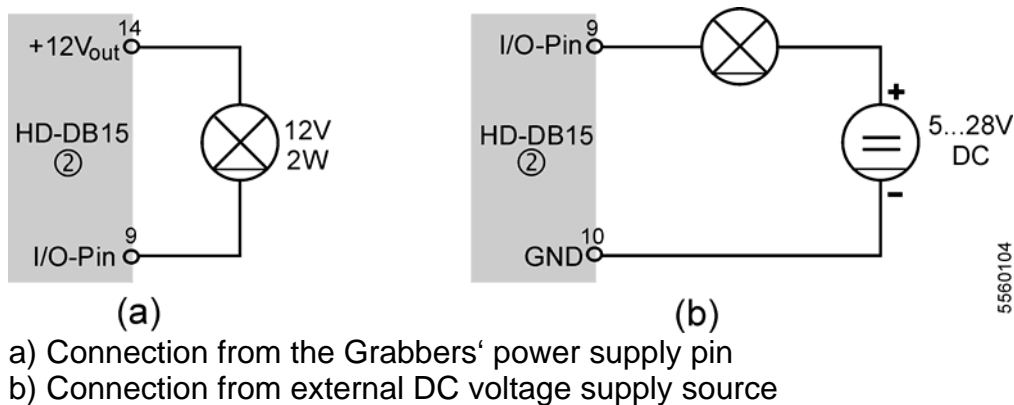


Figure 4: Standard Connections for the I/O Pin as Output

Caution:

The polarity of the connected voltage must be selected in a way that the I/O pin has a constant positive potential.

Negative potential (with reference to Ground) at pin 9 can lead to permanent damage of the framegrabber card!

While using the output function, the connected voltage must be in the range between +5 V and +28 V (I/O pin with reference to Ground). When using the I/O pin as input, the operating voltage must not exceed 28 V.

The I/O pin is not electrically isolated from the video cables, the PC and the other signal lines.

3.5.5 RS6 variant

The RS6 variant contains four relays and a DIP-switch in addition to the standard model.

Relays

The relay outputs are located on the multi-pin connector X14.

multi-pin connector X14	
Pin	Function
1	Relay 1 (N.O.)
2	
3	Relay 2 (N.O.)
4	
5	Relay 3 (N.O.)
6	
7	Relay 4 (N.O.)
8	

Table 5: Relays / multi-pin connector X14

The relays are normal-open – type.

The contacts close, when the corresponding bit is set by software.

Attention:

The relays support loads up to 24 Volt and 1 Ampere.

Minimum contact load: 5V / 1 mA

DIP-switch

Statuses of the DIP-Switch can be read by software. For example they can be used to identify the individual cards in a multi-framegrabber - system. Setting the switches of each framegrabber card to a individual position, the cards – and corresponding inputs – can be identified by the application.

3.5.6 Option Port

The option port provides 12 digital I/O-lines and one I²C-interface to the user. The signals are routed to a connector with 10 x 2 pins. The connector is denoted as X6, pin 1 is located in the lower left. *Figure 5* shows the assignment of the pins.

Note: The current drawn out of pin 1 (+5V) may not exceed 100 mA.

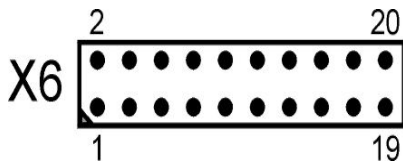


Figure 5: Pin Formation of the Option Port

Option Port, X6					
pin	function	pin	function	pin	function
1	+5V out	8	I/O 6	15	I/O-Pin
2	I/O0	9	I/O 7	16	I/O Clk
3	I/O1	10	I/O 8	17	I ² C SCL
4	I/O2	11	I/O 9	18	I ² C SDA
5	I/O3	12	I/O 10	19	GND
6	I/O4	13	I/O 11	20	GND
7	I/O5	14	N.C.		

Table 6: Pin Assignment for the Option Port

Note for models of the pciGrabber-4plus with -RS6 option installed:

With the -RS6 option installed the I/O8 to I/O11 lines are used internally to control the relays. Thus, the use of these I/Os is in this model limited (see chapter 3.5.5)!

3.5.7 I²C Interface

External devices can be polled or controlled via the I²C interface. In order for this to occur, the external devices must have an I²C interface operating in slave mode.

The I²C interface is available at both the upper and lower HD-DB-15 plugs, and is also available on the internal pin header row of the *Option Port*. It is possible to connect multiple I²C devices to the bus, but these devices must be distinguished by their device addresses. *Table 7* depicts the pin assignments for the HD-DB-15 sockets.

HD-DB-15 ① and ②	
Pin	Function
10	Ground
12	I ² C Bus: SDA
15	I ² C Bus: SCL

Table 7 Connecting the I²C Interface to the Combi Socket

Note:

The maximum cable length is restricted, due to the fact that the I²C interface is driven by TTL signals. For one connected device, depending on the configured transmission rate, the maximum cable length is approx. 1 - 2 m.

Cables with sufficient shielding are to be used to connect this devices.

Information for adapting the I²C interface into application software can be found in *section 7.2.8*, under the functions group „*Transmitting Data via the I²C Interface*“.

3.6 Installing and Starting Up the Framegrabber Card

The framegrabber card converts analog signals from the camera and presents these signals in a digital form to the computer and software.

If you are not familiar with insertable cards, please take the time to familiarize yourself with the instructions and equipment. The following tasks are not difficult, but must be done with caution.

3.6.1 Installing the Framegrabber Card

Caution:

The computer must be disconnected from the power supply. Please ensure that the device does not have any power supplied to it.

- Remove the housing of the PC (normally screwed).
- Select a free PCI slot
(PCI slots are usually the short white parallel slots on the motherboard). Please ensure that the selected PCI slot has busmaster capabilities.

At most motherboards all of the PCI slots have busmaster capabilities. Slave slots are usually labeled accordingly.

If you are unsure whether the slot is a master slot or not, *please refer to the computer's mother board's User's Manual to obtain more information.*

Caution:

If the pciGrabber-4plus is installed into a slave slot, it is possible that the system will no longer start-up (boot). In any case, the pciGrabber-4plus will not function correctly.

- Remove the slot cover from the PC housing. The slot cover is located in front of the selected slot (unscrew or break off).
- As shown in *Figure 6*, insert the pciGrabber-4plus into the slot with the connectors facing outwards. The card should be inserted securely.
- Do not force the card into the slot. Forcing the card into the slot can damage the mother board, as well as the card.
- Ensure that the framegrabber card is inserted into the right PCI slot. Line up the golden contact strips with the PCI slot's receptacle. Some resistance will be encountered as the contact strips spread apart the contact springs.

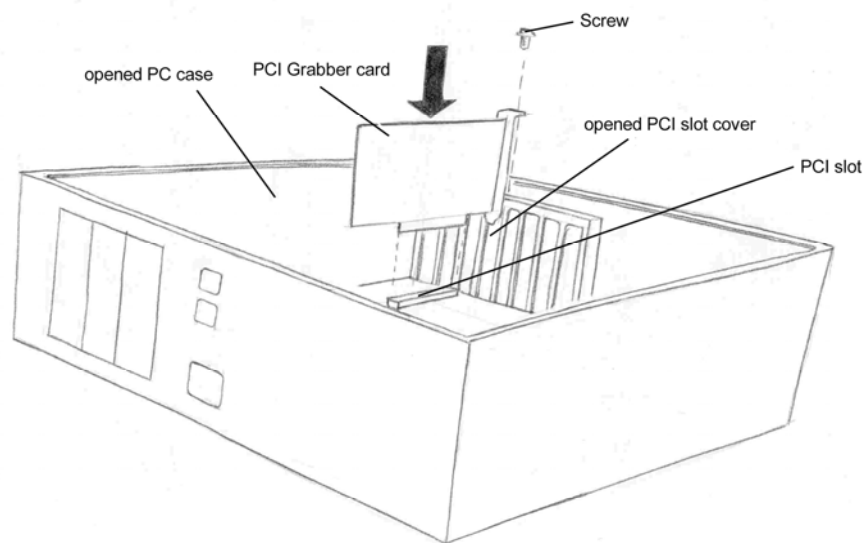


Figure 6: Inserting the Card into the PCI Slot

- After inserting the card, please ensure that the Grabber card fits snugly into the receptacle and that there is no interference from neighboring contacts.

The pciGrabber-4plus is intended for use with 5 V PCI bus systems. An encoded notch on the PCI slot ensures that framegrabber cards cannot be installed in 3.3 V systems.

Caution:

For stability reasons and to ensure a secure ground connection to the computer's housing, screw the card to the housing (*see Figure 6*).

-
- One of the computer's 3¹/₂" power plugs has to be connected to the power inlet of the framegrabber, if the camera power supply feature is intended to be used. (The 1.6 A fuse in socket F2 secures the power supply output.) *Figure 7* depicts the connection.

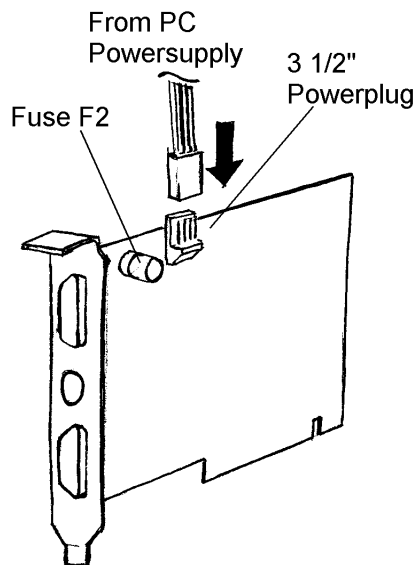


Figure 7: Set up of the power supply feature

- Close the computer's housing.

3.7 Connecting Video Sources

It is possible to connect one or more video sources to the pciGrabber-4plus (*see Figure 8*). These sources can either be video cameras, video recorders or any other video source with appropriate outputs (composite or *S-Video*).

Depending on the framegrabber model, both 3 composite and one S-Video (VD-009-X1 and VD-011), or 9 composite and one S-Video source (VD-009) can be connected to the framegrabber.

Changing channels occurs via software, for example the demo application that is shipped with the framegrabber.

Only one input can be active and digitized at a time.

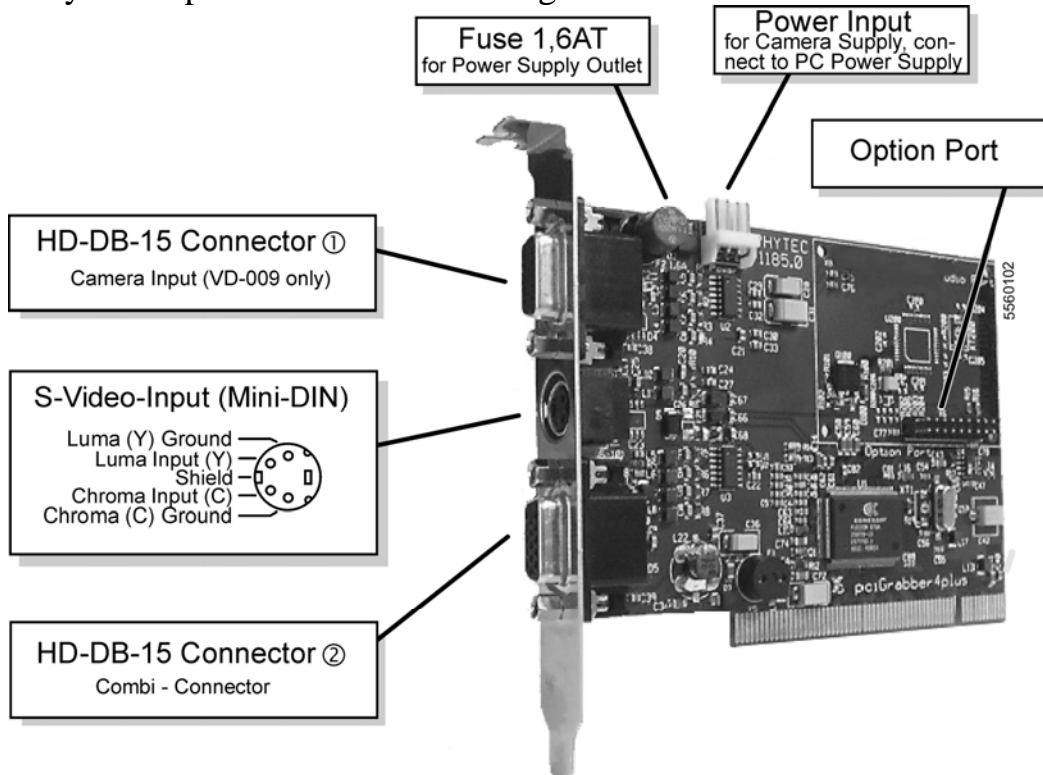


Figure 8: Overview of the pciGrabber-4plus Connectors

The composite inputs are located on the 15-pin HD-DB sockets (① and ②). In addition to the composite video inputs, a +12V power output for supplying a camera is available at the second 15-pin HD-DB socket ② (when the 3¹/₂“ power plug is connected internally).

Necessary cables can be ordered from PHYTEC. Please refer to section 3.2, “Accessories“.

Note:

The second HD-DB-15 socket ② is also referred to as the ‘COMBI’ socket.

Start-Up
pciGrabber-4plus

An S-Video signal can also be applied to the Mini DIN socket. Or, alternatively, special cables can be used to connect S-Video sources to the second HD-DB-15 socket ②.

The Video Power Combi cable (WK-075) is offered by PHYTEC exclusively and enables connection of an S-Video camera. Besides the video input, the cable integrates also the power supply for the camera. Thus, this type of connection replaces an external power supply.

Spare fuses for the camera power supply can also be ordered from PHYTEC (*see section 3.2*).

Additional information for the pin assignments of the sockets can be found in the section *Technical Data*.

3.7.1 Video Connections

Various video source connections for the framegrabber are briefly described in this section.

All of the pictured cables can be ordered from PHYTEC. The illustration of the cables includes a brief cable description and the PHYTEC part number (*see the figure below*).

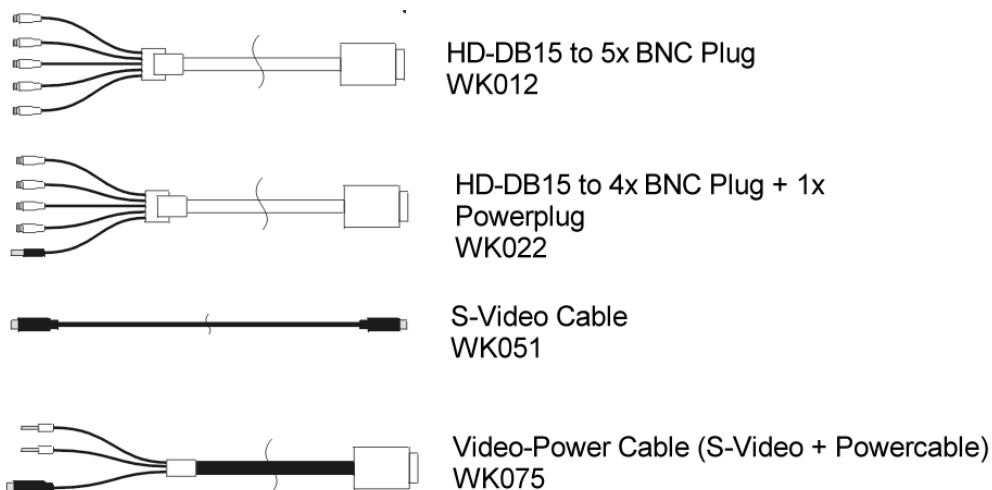


Figure 9: Video Connector Cables - (Description and PHYTEC part number)

For more information on compatibility, please refer to the video source's data sheets.

Connection options may vary according to the framegrabber model.

The following images categorize the various framegrabber models:

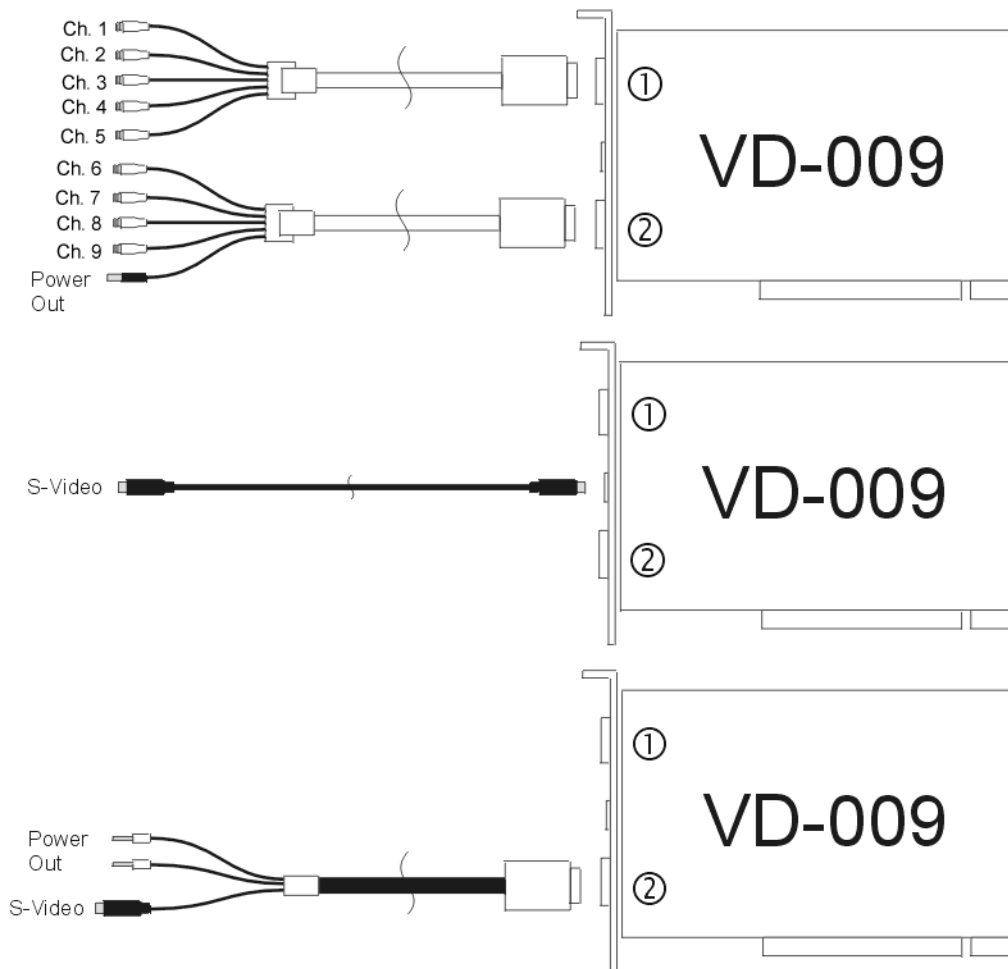


Figure 10: Connectors for the VD-009 Model

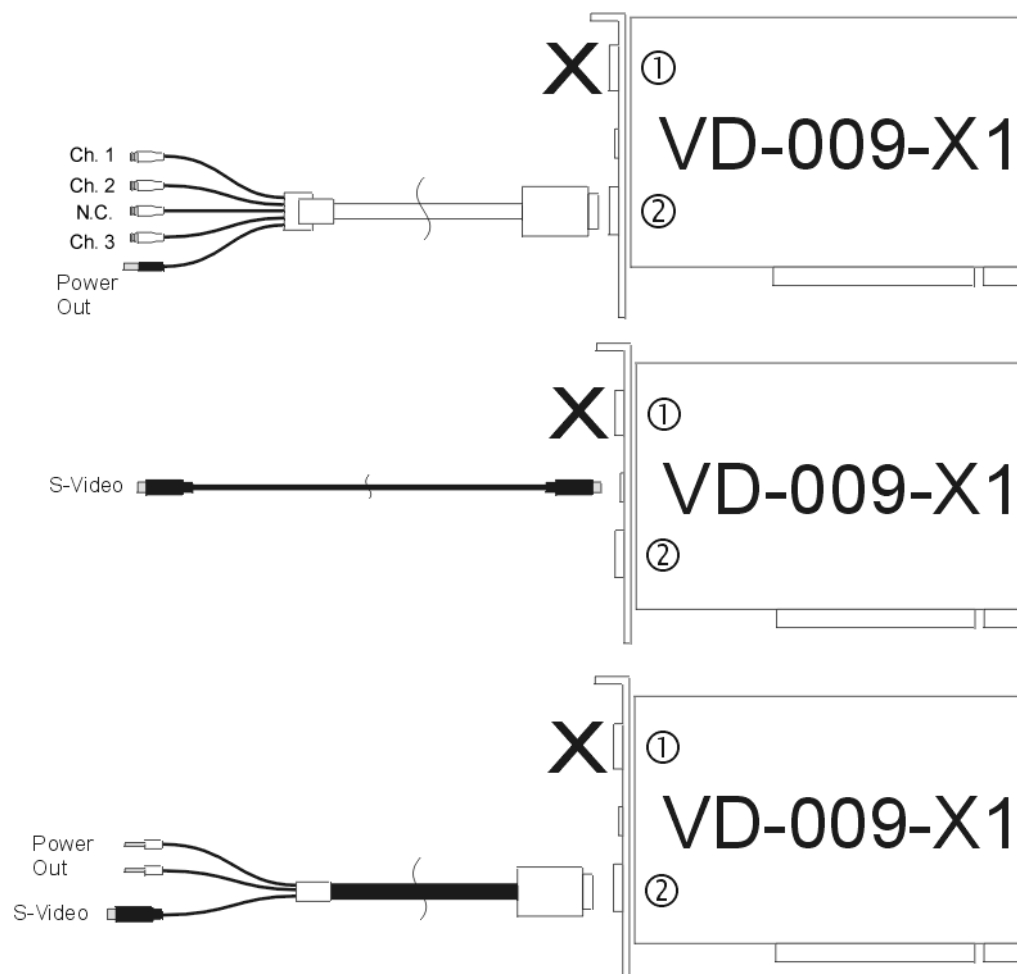


Figure 11: Connectors for the VD-009-X1 Model

The following section briefly describes the above depicted cables.

3.7.2 The Video/Power Cable

Figure 12 depicts the connection of a video/power cable to a camera (i.e. PHYTEC VCAM 110-x)

The video/power cable is intended for connection of an S-Video camera. A dual wire with open ends integrated into the cable functions as a power supply for the camera by the framegrabber (red= +12 V, black = Ground).

The HD-DB-15 plug of the cable is connected to the framegrabber's HD-DB-15 connector ②. Socket ② must be used since the power supply is located at pin 14.

Caution:

To avoid short circuits during installation, establish the connection to the camera first before connecting the cable to the framegrabber card! We recommend to establish any connection while the computer is off.

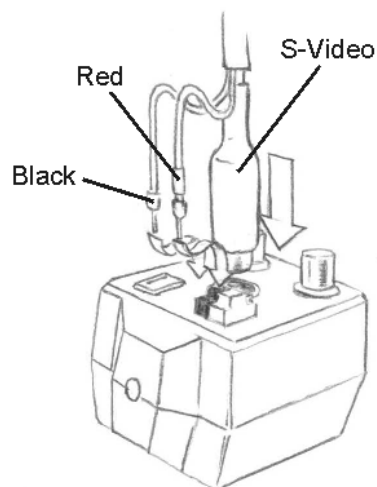


Figure 12: Connecting a Camera (VCAM 110-x) to the Video Power Cable (example)

3.7.3 The S-Video Cable

The S-Video cable is connected to the framegrabber using the round Mini DIN socket. The video source (i.e. camera with S-Video output) should have a similar connector.

3.7.4 The Composite Connectors

It is possible to connect the composite outputs (BNC plug) with a video source using a BNC plug.

Note:

If the composite sources contain a RCA connector a RCA/BNC adapter (75 Ω) must be used.

The end that contains the HD-DB15 socket is inserted into the framegrabber.

Depending on the design of the cables, it is possible to supply either 5 composite sources, or 4 composite sources and a power supply (pin 14 on the framegrabber socket ②).

Caution:

The cable containing five BNC plugs may only be connected to socket ① of the framegrabber. This cable is only suitable for type VD-009 and must not be used together with VD-009-X1.

In order to digitize an image, the correct channel must be selected in the user's software and in the demo program. It is possible for the included software to automatically recognize which channel is supplied with a signal (*see section 6*).



Now please proceed to *chapter Fehler! Verweisquelle konnte nicht gefunden werden.* to install the driver software for Microsoft Windows and to start up the framegrabber with the demo application.

4 pciGrabber-4express

4.1 Scope of delivery (pciGrabber-4express)

- the PCI express-card
- Installation CD with
 - Demo software (Windows 95/98/ME/XP, NT4.0, 2000 and Windows VISTA)
 - Driver library for DOS (with DOS4GW)
 - Driver software for Windows‘ 95/98/ME/XP, NT4.0, 2000 and Windows VISTA
 - Twain driver for applications with Twain interfaces
- this pciGrabber-4plus/ pciGrabber-4express manual

4.2 Accessories

The following pciGrabber-4express accessories may be ordered from PHYTEC:

- Composite connector cable for four cameras and a power supply output (12 VDC) for a camera (lower socket of VD-009-x). HD-DB15 to 4 x BNC-plug and 1 x power plug, length approx. 2 m – part number WK022
- S-Video connector cable for connection of color cameras with a 4-pin Mini-DIN plug (S-Video output). Length, approx. 2 m – part number: WK051
- BNC connector cable for connection of cameras with BNC-connector. Part number: WK058 (2 m) or WK039 (10 m)
- Combi connector cable for color cameras with S-Video connection and 12 V power supply. HD-DB15 to 1 x Mini DIN plug and 1 x power supply (open ends). Compatible for example with the VCAM 110, 120. Length approx. 2 m. – part number WK075.
- Replacement fuse 1.6A slow blow TR5 for camera power supply (receptacle F2) – part number KF012
- Replacement fuse 500mA slow blow TR5 for camera power supply (receptacle F1) – part number KF014

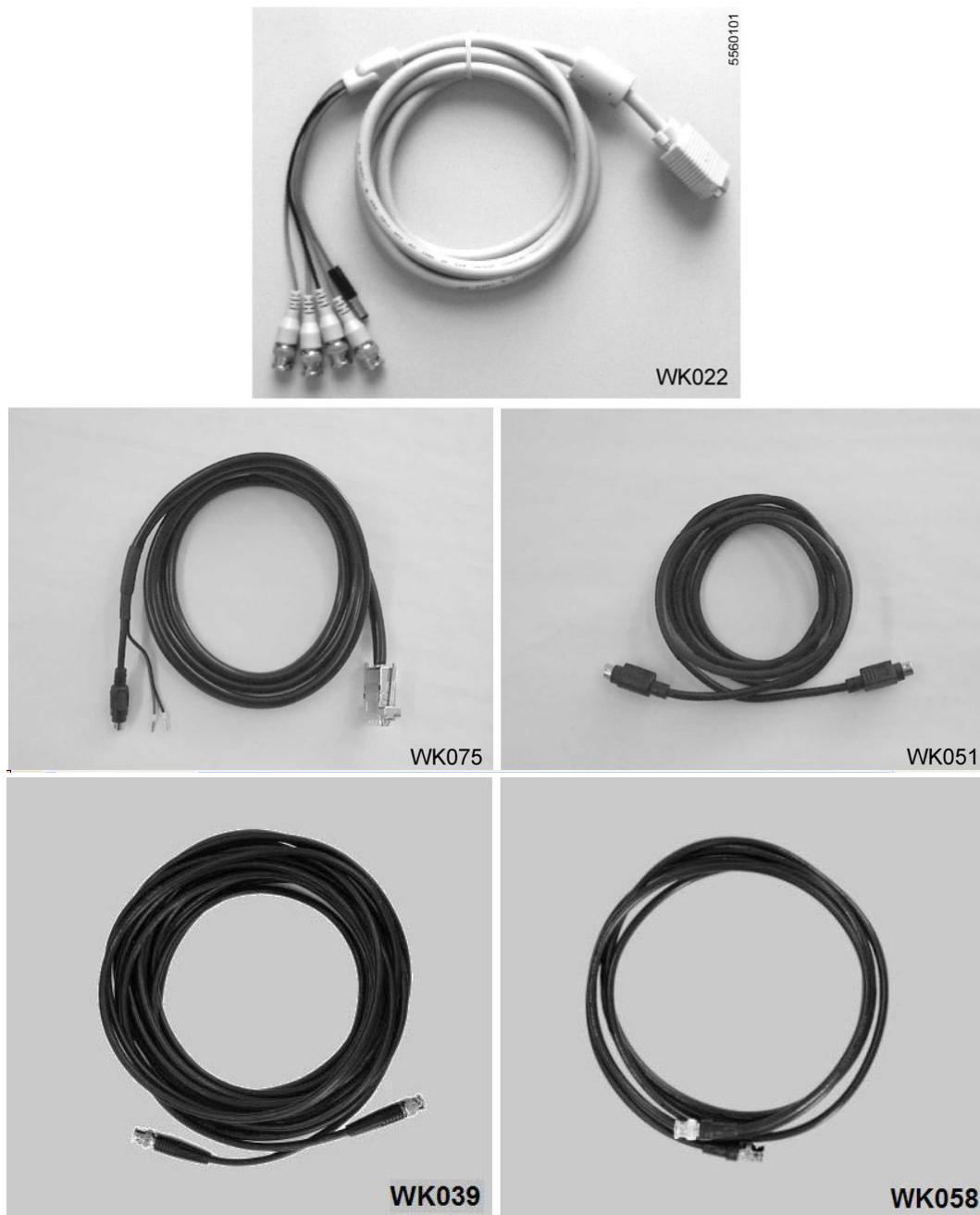


Figure 13: Accessory Cables pciGrabber-4express

4.3 Technical Data VD-011

Physical

Dimensions:	120 x 95 x 19 mm plus backplate and slot
Data Bus:	x1 PCI Express-Bus (PCI Express Base Spec. Rev. 1.0a compliant)
Power Supply:	+3.3V V (250 mA idle, 300 mA digitizing) (from PCI Express-Bus)
Inputs:	<u>Model VD-011:</u> 3 composite video inputs, 75 Ω , 1 V _{ss} ¹ 1 S-Video input 75 Ω (0.7 V _{ss} / 0.3 V _{ss})
Video Format:	PAL (B,G,H,I), NTSC (M) or corresponding CCIR monochrome format
Synchronization:	Composite sync. or sync to Y-signal external synchronization is not possible
Data Format:	16 Mio. colors: RGB32, RGB24, YcrCb 4:2:2, YcrCb 4:1:1 64,000 colors: RGB16 32,000 colors: RGB15 256 gray shades: Y8 gray scale

¹: If an S-Video input is not being used, an extra composite input is available

Image

Resolution: maximum 768 x 576 pixels (PAL)
or 640 x 480 pixels (NTSC)
Resolution is freely scalable in X and Y directions
up to 14:1

Image Transfer

Rate: field 20 ms (one odd or even field)
full frame 40 ms
Image transfer to the main memory in real time
(Bus master transfer)

Used

Resources: 4 kByte main memory (register field)
/INTA

Image control: Gamma correction (selectable)
Brightness (+/- 50 %)
Contrast (0 % ... 235 %)
Color saturation (U: 0...201 %, V: 0...283 %)
Hue (+/- 90°, only with NTSC)

Image Storage: 630 Byte FIFO on-board,
Real time storage in the PC main memory
Even-/odd field memory separated or
Common full frame memory (selectable)

Ports: 12-bit parallel I/O, TTL signal (multi-purpose)

Parameter	Symbol	Min	Max
Input High Voltage	V_{IH}	2,0 V	5 V
Input Low Voltage	V_{IL}	-0,5 V	0,8 V
Output High Voltage	V_{OH}	2,4 V	-
Output Low Voltage	V_{OL}	-	0,4 V
Input Low Current	I_{IL}	-	-70 μ A
Input High Current	I_{IH}	-	70 μ A

1 I/O Port (driven transistor, 28 V/0.8 A_{max})

Parameter	Symbol	Min	Max
Input High Voltage	V_{IH}	2,0 V	28 V
Input Low Voltage	V_{IL}	-0,5 V	0,5 V
Output High Voltage	V_{OH}	5 V	25 V
Output Low Voltage	V_{OL}	0 V	1,4 V
Input Low Current	I_{IL}	-	-700 μ A
Input High Current	I_{IH}	-	70 μ A
Output HiZ Current	I_{OZ}	-	500 μ A
Output On Current	I_{OON}	-	800 mA
Switching frequency	f_{IO}		200 Hz

1 I²C interface (Master)

Parameter	Symbol	Min	Max
Transmission rate ¹	f_{I2C}	99,2 kHz	396,8 kHz
Input High Voltage	V_{IH}	3,5 V	5 V
Input Low Voltage	V_{IL}	-0,5 V	1,5 V
Hysteresis	V_{hys}	0,2 V	
Input High Current	I_{IH}	-	10 μ A
Input Low Current	I_{IL}	-	-10 μ A
Output Low Voltage	V_{OL}	-	0,4 V

Relays: (version -RS6 only)
4 Relay-outputs
(N.O., 24V, 1A max)

DIP-Switch: (version -RS6 only)
quadruple DIP-switch

¹: frequencies can be selected software

Connectors:	<u>Model VD-011</u>	
	HD-DB-15 Socket:	4 Composite video inputs 1 S-Video chroma input 1 I ² C interface 1 I/O connection, driven 1 output for camera power supply, +12 V/1.5 A _{max}
	Mini DIN socket:	S-Video input
	BNC sockets:	2 Composite video inputs
	Pin header row 2 x 10: (not on backplate)	GPIO port, 12 x TTL I/O I ² C interface I/O connection, driven
	Pin header row 2 x 4: (not on backplate)	4 Relay-outputs (optional only VD-011-RS6)

4.4 Addresses and Resources

The pciGrabber-4express uses a range of 4 kBytes in the main memory of the PC for the local registers. The addressing region is automatically specified by the BIOS and no hardware setup (jumper setting) is required.

Several pciGrabber-4express can be installed in one system. The boards are configured automatically by the BIOS for different addresses.

It is not possible to determine which board is configured to which address. The base address of each board can be obtained by the PCI-BIOS. For the pciGrabber-4express the driver software determines the address via the BIOS and defines a device number. The driver also can determine the number of boards within the system and is able to control each board by its particular device number.

It is not possible to determine which board will be specified by which device number. This will be determined by the PCI-BIOS and the architecture of the PC-motherboard. Usually the addresses are allocated in sequence of the numbering of the PCI slots. This might deviate for different manufacturers. To solve that problem the RS6 version can be used. It allows to give every framegrabber an explicit address by different DIP-switch settings.

The pciGrabber-4express will activate an interrupt in case of certain events or a distinct operational status.

The framegrabber is a *single function device* so only the interrupt line /INTA of the PCI-bus is used. To this PCI-bus-interrupt an interrupt of the PC is allocated by the BIOS, so that the program can react to this event.

The source of the interrupt can be determined from the interrupt status register of the framegrabber.

Several boards can trigger the same interrupt /INTA, so it has to be determined which board caused the interrupt.

4.5 Socket Pinout

Note:

The following description of the framegrabber's connectors is intended as a technical reference.

4.5.1 Composite Inputs

All composite video sources with an output level of $1 V_{SS}$ and an impedance of 75Ω can be used. For more information on video standards, please refer to *section 4.3*

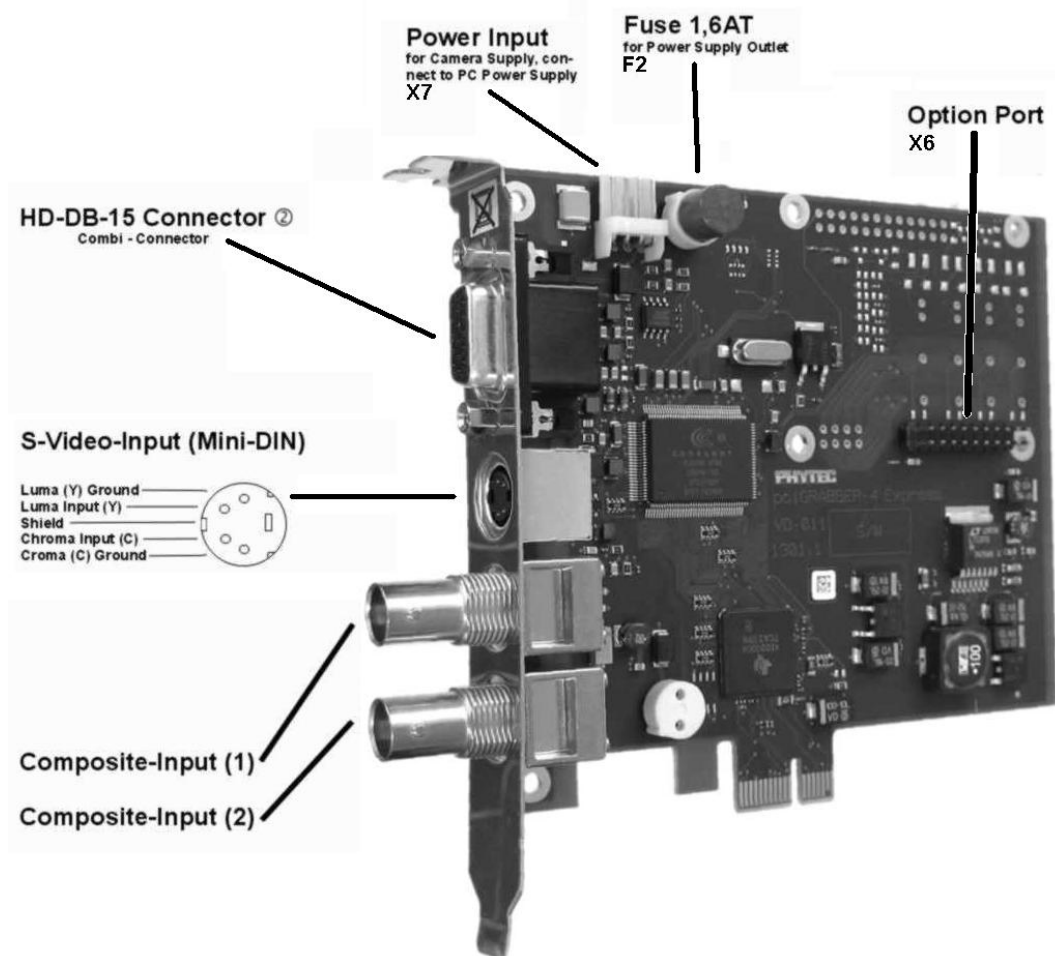


Figure 14: Connectors of the pciGrabber-4express

- **Version VD-011**

Three composite inputs available to the framegrabber are located on the HD-DB-15 socket ②. Additional two of the three composite inputs are also located on BNC-plugs (refer to *Figure 14*).

The input assignment for the channel numbers is depicted below:

pciGrabber-4express (VD-011)

BNC (1)	
Pin	Function
Tip	Composite Input 1
Ring	Signal Ground

BNC (2)	
Pin	Function
Tip	Composite Input 2
Ring	Signal Ground

HD-DB-15 ② (X2)	
Pin	Function
1	Composite Input 1
2	Composite Input 2
3	S-Video: Luma
4	S-Video: Chroma
5	Signal Ground
6	Signal Ground
7	Signal Ground
8	Signal Ground
9	I/O-Pin
10	Pwr Supply Ground(-)
11	Signal Ground
12	I ² C Bus: SDA
13	Composite Input 3
14	+12 V out (camera supply)
15	I ² C Bus: SCL

Table 8: Pin Assignments of the pciGrabber-4express (VD-011)

In addition to the composite video inputs, a power output is available. The power output enables a +12 V power source from the host PC to be connected to a camera. *Refer to section 4.5.3, „Power Supply Output“.*

PHYTEC offers connecting cables that enable the application of the video signal via BNC plugs.

Caution:

If the power output is not intended for use, fuse F1 or F2 can be removed, in order to avoid a power supply of +12 V to be present at connector ②.

4.5.2 S-Video Connection

The advantage of this design is the separate conduct of brightness and color signal. This prevents disturbing Moiré effects for fine image structures and improves the resolution of the color image.

There are two possibilities for connecting an S-Video source to the pciGrabber-4express:

- Direct an S-Video signal to the 4-pin Mini DIN socket (X3). The socket is wired corresponding to the S-Video standard (*refer to Figure 14*). The connection of the camera is made by a standard S-Video cable..
- Using a special cable (i.e. WK075), it is possible to connect an S-Video camera to the HD-DB-15 socket ②. Connecting the S-Video camera in this manner allows additional power supplies of +12 V, or additional signals, to be directed by the same connection cable. If such a cable is being used, then the following pins are to be connected (connection of the power supply is optional):

HD-DB-15 ② (X2)	
Pin	Function
4	S-Video: Chroma
5	Signal Ground
6	Signal Ground
10	Pwr Supply Ground(-)
13	S-Video: Luma
14	+12 V out (Camera supply)

Table 9 Connection of the S-Video Input to the HD-DB-15 Socket

Caution:

Both S-Video inputs can **not** be connected at the same time. Either the Mini DIN input or the HD-DB-15 socket ② can be used. The user must select in the application software which socket is connected to the S-Video source. (Automatic selection of the socket is also possible).

4.5.3 Power Supply Output

The *pciGrabber-4express* provides a power supply output of +12 V at pin 14 on the lower HD-DB-15 socket, ② for the connected camera(s). Therefore, a supplemental power supply is not necessary if the camera is installed in the vicinity of the PC. The maximum current load is 1.5 A

Note:

In order to use the power supply output, the *pciGrabber-4express* must be connected to the PC power supply. Connect a free power supply cable from the power supply to the connector plug (X7), located on the framegrabber. 3,5“ floppy drive power supply connectors are suitable for use.

The +12V voltage supplied to the plug X7 is provided at the HD-DB-15 plug ② from the framegrabber. For more information on installing the power supply cables, *refer to section 4.6.1.*

A miniature fuse, F2, protects the output. Additional replacement fuses can be ordered from PHYTEC (part number KF012).

Regarding the output current, please adhere to the output power supply (+12V) specifications of the PC power supply.

4.5.4 I/O Pin

A universal I/O pin is located at pin 9 of the HD-DB-15 plug ②. This I/O pin is universal in the sense that it can be used as either input or output. In order to use this I/O pin, ensure that the application software supports this function.

Using the I/O pin as input enables the application software to receive control signals. The input can be polled from the program and is not connected to any special functions.

When using the function as output, the application software is able to convey control signals to other devices. The output can be set to Hi-Z state or set to ground connection by the software.

I/O Pin Functions

- **Input**

An external voltage (with reference to Ground) can be connected to the I/O pin. If the voltage is between 0 V and 0.5 V, then the program reads a „0“. If the voltage is between 2 V and 28 V, then the logic signal level is set to „1“. The positive connection must be at terminal 9 (see *Table 10*).

HD-DB-15 ② (X2)	
Pin	Function
9	I/O Pin (+)
10	Ground (-)

Table 10: Connection of the I/O Pin to the Combi Socket

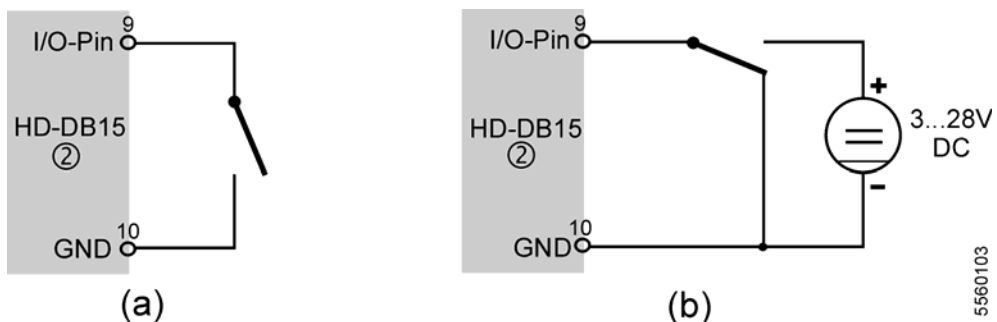


Figure 15: Standard Connections for the I/O Pin as Input

- **Output**

If the pin is used as an output, it has the following behavior:

If the software sets the pin to logic „1“ output, then pin 9 is connected to Ground (i.e. pin 10), via a transistor.

If the software sets the pin to logic „0“, then the transistor is in high impedance (Hi-Z) - state and thus, there is no connection between pin 9 and Ground.

In order to use the output, an external power supply, in the range of 5 V and 28 V, is required. It is also possible to use the power supply pin (pin 14) for power supply. *Figure 16* depicts two possible connections.

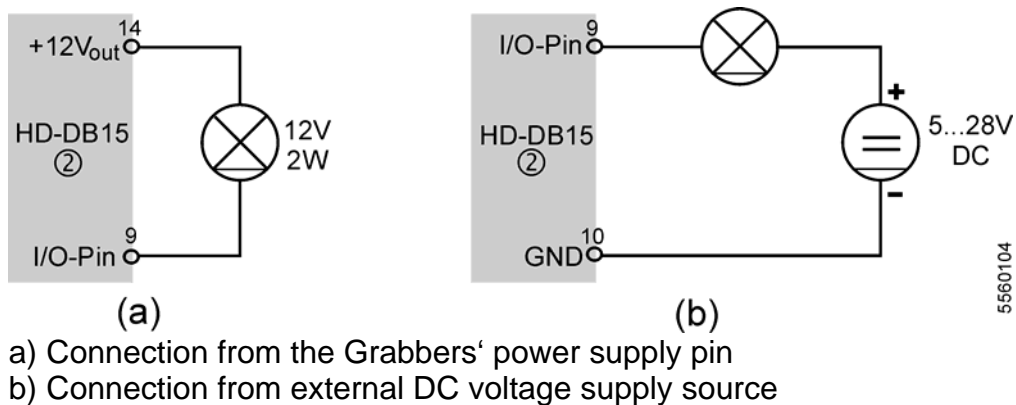


Figure 16: Standard Connections for the I/O Pin as Output

Caution:

The polarity of the connected voltage must be selected in a way that the I/O pin has a constant positive potential.

Negative potential (with reference to Ground) at pin 9 can lead to permanent damage of the framegrabber card!

While using the output function, the connected voltage must be in the range between +5 V and +28 V (I/O pin with reference to Ground). When using the I/O pin as input, the operating voltage must not exceed 28 V.

The I/O pin is not electrically isolated from the video cables, the PC and the other signal lines.

4.5.5 RS6 variant

The RS6 variant contains four relays and a DIP-switch in addition to the standard model.

Relays

The relay-outputs are located on the multi-pin connector X14.

multi-pin connector X14	
Pin	Function
1	Relay 1 (N.O.)
2	
3	Relay 2 (N.O.)
4	
5	Relay 3 (N.O.)
6	
7	Relay 4 (N.O.)
8	

Table 11: Relays / multi-pin connector X14

The relays are normal-open – type.

The contacts close, when the corresponding bit is set by software.

Attention:

The relays support loads up to 24 Volt and 1 Ampere.

Minimum contact load: 5V / 1 mA

DIP-switch

Statuses of the DIP-Switch can be read by software. For example they can be used to identify the individual cards in a multi-framegrabber - system. Setting the switches of each framegrabber card to a individual position, the cards – and corresponding inputs – can be identified by the application.

4.5.6 Option Port

The option port provides 12 digital I/O-lines and one I²C-interface to the user. The signals are routed to a connector with 10 x 2 pins. The connector is denoted as X6, pin 1 is located in the lower left. *Figure 17* shows the assignment of the pins.

Note: The current drawn out of pin 1 (+5V) may not exceed 100 mA.

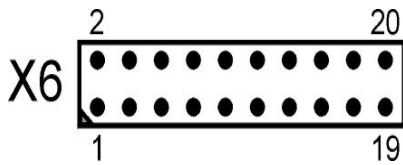


Figure 17: Pin Formation of the Option Port

Option Port, X6					
pin	function	pin	function	pin	function
1	+5V out	8	I/O 6	15	I/O-Pin
2	I/O0	9	I/O 7	16	I/O Clk
3	I/O1	10	I/O 8	17	I ² C SCL
4	I/O2	11	I/O 9	18	I ² C SDA
5	I/O3	12	I/O 10	19	GND
6	I/O4	13	I/O 11	20	GND
7	I/O5	14	N.C.		

Table 12: Pin Assignment for the Option Port

Note for models of the pciGrabber-4express with -RS6 option installed:

With the -RS6 option installed the I/O4 to I/O7 lines are used internally to read out the DIP switches. Thus, the use of these I/Os is in this model limited (see chapter 4.5.5)!

4.5.7 I²C Interface

External devices can be polled or controlled via the I²C interface. In order for this to occur, the external devices must have an I²C interface operating in slave mode.

The I²C interface is available at the HD-DB-15 connector and also at the internal pin header row of the *Option Port*. It is possible to connect multiple I²C devices to the bus, but these devices must be distinguished by their device addresses. *Table 13* depicts the pin assignments for the HD-DB-15 sockets.

HD-DB-15 ① and ②	
Pin	Function
10	Ground
12	I ² C Bus: SDA
15	I ² C Bus: SCL

Table 13 Connecting the I²C Interface to the Combi Socket

Note:

The maximum cable length is restricted, due to the fact that the I²C interface is driven by TTL signals. For one connected device, depending on the configured transmission rate, the maximum cable length is approx. 1 - 2 m.

Cables with sufficient shielding are to be used to connect this devices.

Information for adapting the I²C interface into user software can be found in *section 7.2.8*, under the functions group „*Transmitting Data via the I²C Interface*“.

4.6 Installing and Starting Up the Framegrabber Card

The framegrabber card converts analog signals from the camera and presents these signals in a digital form to the computer and software.

If you are not familiar with insertable cards, please take the time to familiarize yourself with the instructions and equipment. The following tasks are not difficult, but must be done with caution.

4.6.1 Installing the Framegrabber Card

Caution:

The computer must be disconnected from the power supply. Please ensure that the device does not have any power supplied to it.

- Remove the housing of the PC (normally screwed).
- Select a free PCI Express slot
(PCI Express slots are parallel slots on the motherboard, often marked yellow).
If you are unsure whether the slot is a PCI Express slot or not, please refer to the computer's motherboard's User's Manual to obtain more information.
- Remove the slot cover from the PC housing. The slot cover is located in front of the selected slot (unscrew or break off).
- As shown in *Figure 18*, insert the pciGrabber-4express into the slot with the connectors facing outwards. The card should be inserted securely.
- Do not force the card into the slot. Forcing the card into the slot can damage the mother board, as well as the card.
- Ensure that the framegrabber card is inserted into the right PCI Express slot. Line up the golden contact strips with the PCI Express slot's contacts. Some mechanical resistance will be encountered as the contact strips spreads apart the contact springs.

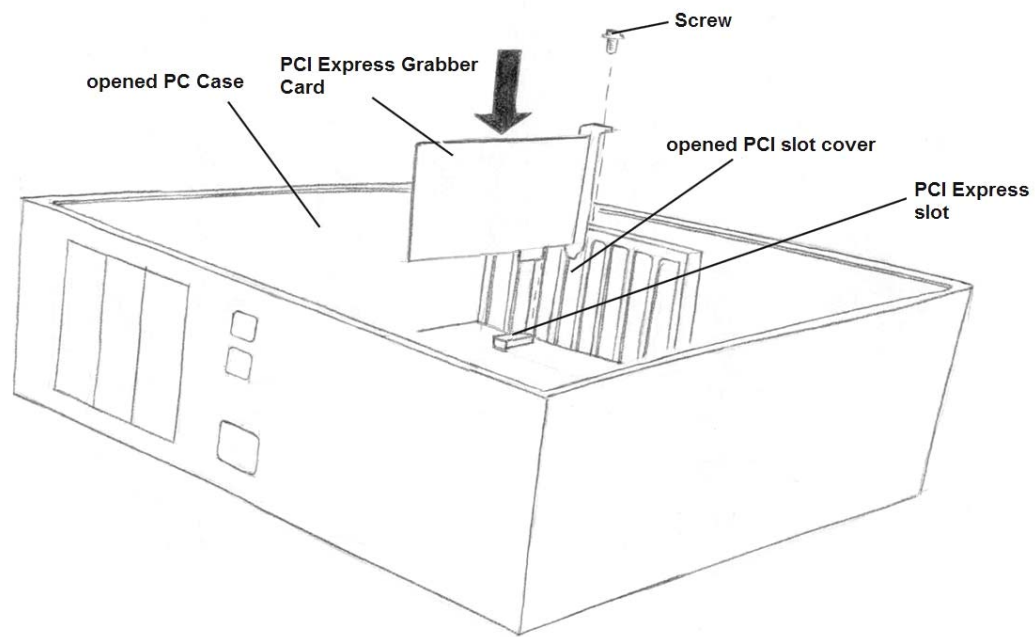


Figure 18: Inserting the Card into the PCI Express- Slot

- After inserting the card, please ensure that the framegrabber card fits snugly into the receptacle and that there is no interference from neighboring contacts.

Caution:

For stability reasons, and to ensure a secure Ground connection to the computer's housing, screw the card to the housing (see Figure 18).

- One of the computer's 3¹/₂" power plugs has to be connected to the power inlet of the framegrabber, if the camera power supply feature is intended to be used. (The 1.6 A fuse in socket F2 secures the power supply output. *Figure 19* depicts the connection.

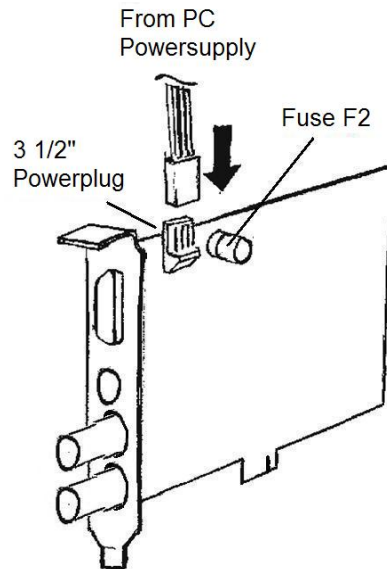


Figure 19: Set up of the power supply feature

- Close the computer's housing.

4.7 Connecting Video Sources

It is possible to connect one or more video sources to the pciGrabber-4express (see *Figure 20*). These sources can either be video cameras, video recorders or any other video source with appropriate outputs (composite or *S-Video*).

The pciGrabber-4express, has 3 composite and one S-Video Inputs.

Changing channels occurs via software, for example the demo application that is shipped with the framegrabber.

Only one input can be active and digitized at a time.

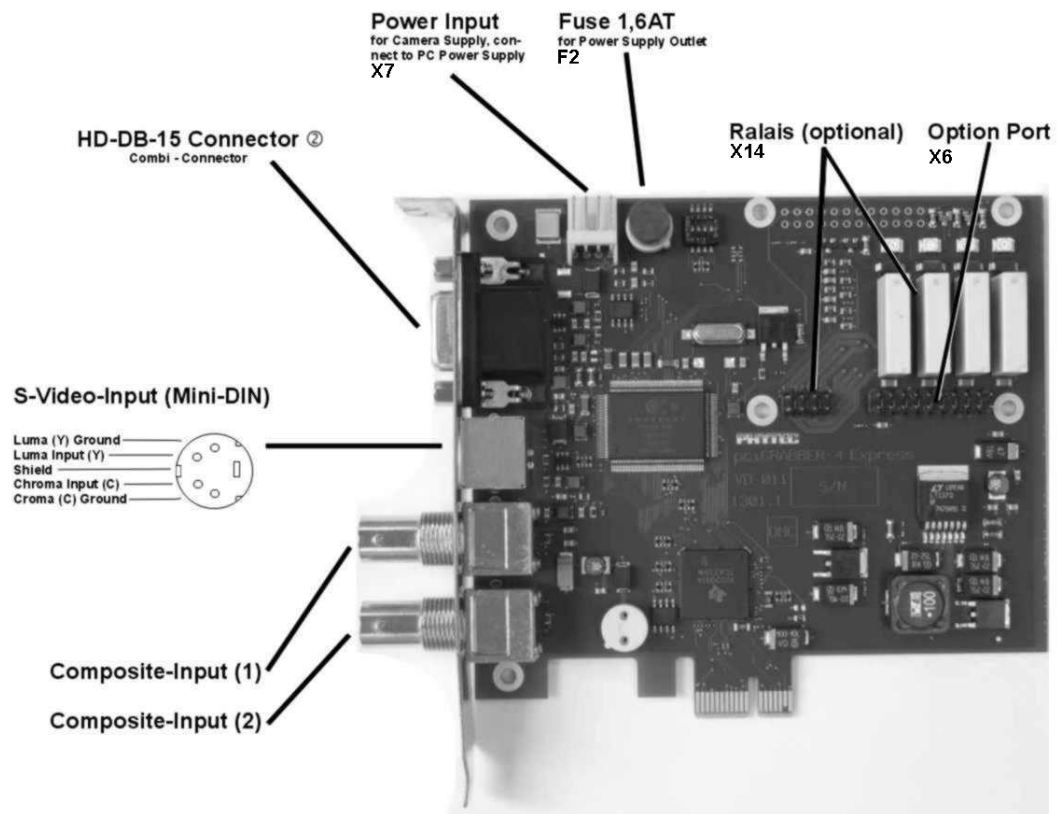


Figure 20: Overview of the pciGrabber-4express Connectors

The composite inputs are located on the 15-pin HD-DB socket ② and also on two BNC-sockets. In addition to the composite video inputs, a power output for supplying a camera is available at the 15-pin HD-DB socket ② (when the 3¹/₂“ power plug is connected internally).

Necessary cables can be ordered from PHYTEC. Please refer to section 4.2, “Accessories“.

Note:

The HD-DB-15 socket ② is also referred to as the COMBI socket.

An S-Video signal can also be applied to the Mini DIN socket. Or, alternatively, special cables can be used to connect S-Video sources to the HD-DB-15 socket ②.

The Video Power Combi cable (WK-075) is offered by PHYTEC exclusively and enables connection of an S-Video camera.

Besides the video input, the cable integrates also the power supply for the camera. Thus, this type of connection replaces an external power supply.

Spare fuses for the camera power supply can also be ordered from PHYTEC (*see section 4.2*).

Additional information for the pin assignments of the sockets can be found in the section entitled *Technical Data*.

4.7.1 Video Connections

Various video source connections for the framegrabber are briefly described in this section.

All of the pictured cables can be ordered from PHYTEC. The illustration of the cables includes a brief cable description and the PHYTEC order number (*see the figure below*).

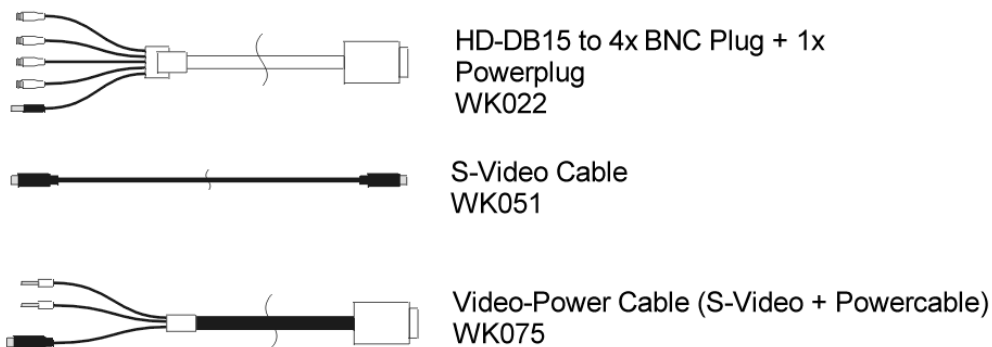
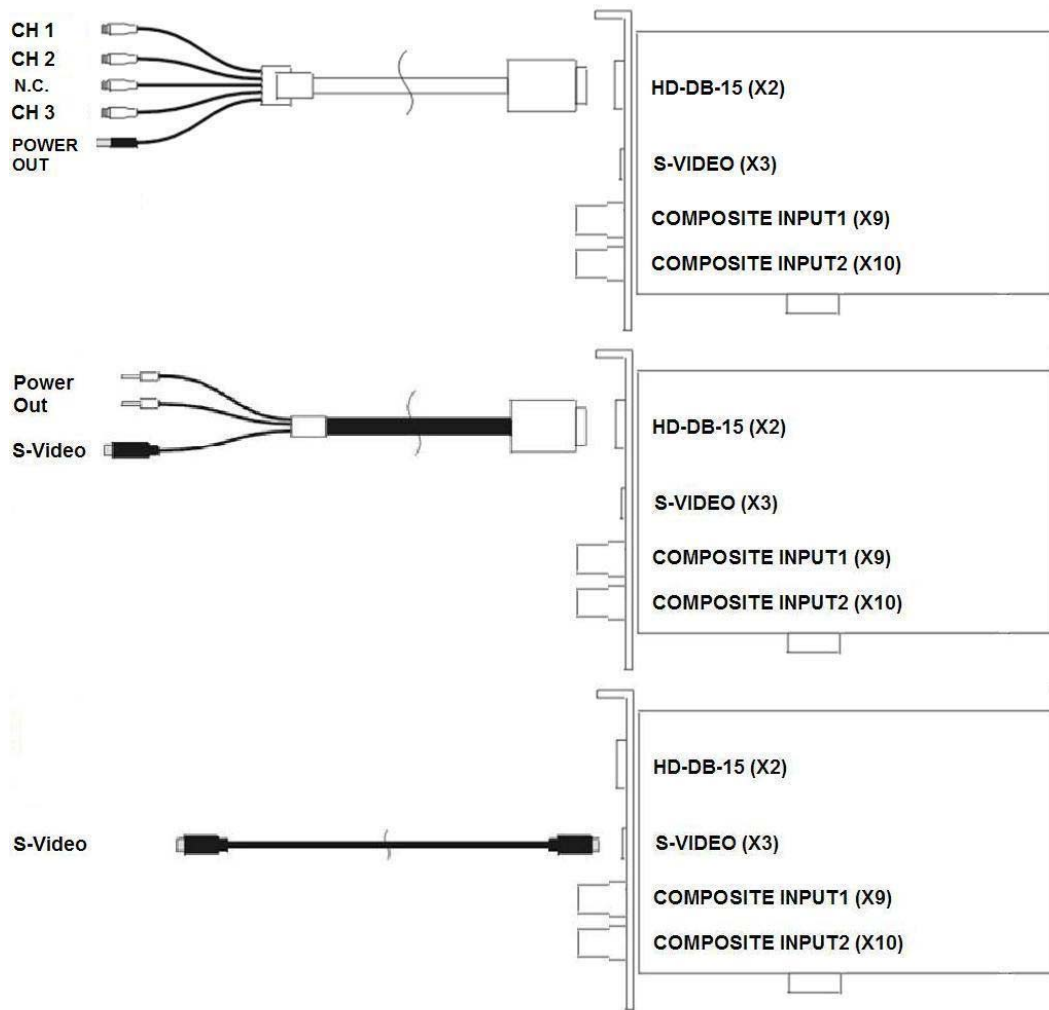


Figure 21: Video Connector Cables - (Description and PHYTEC part number)

For more information on compatibility, please refer to the video source's data sheets.

Connection options may vary according to the framegrabber model.

The following images categorize the various framegrabber models:



Start-Up

pciGrabber-4express

Figure 22: Connectors for the VD-011 Model (part 1)

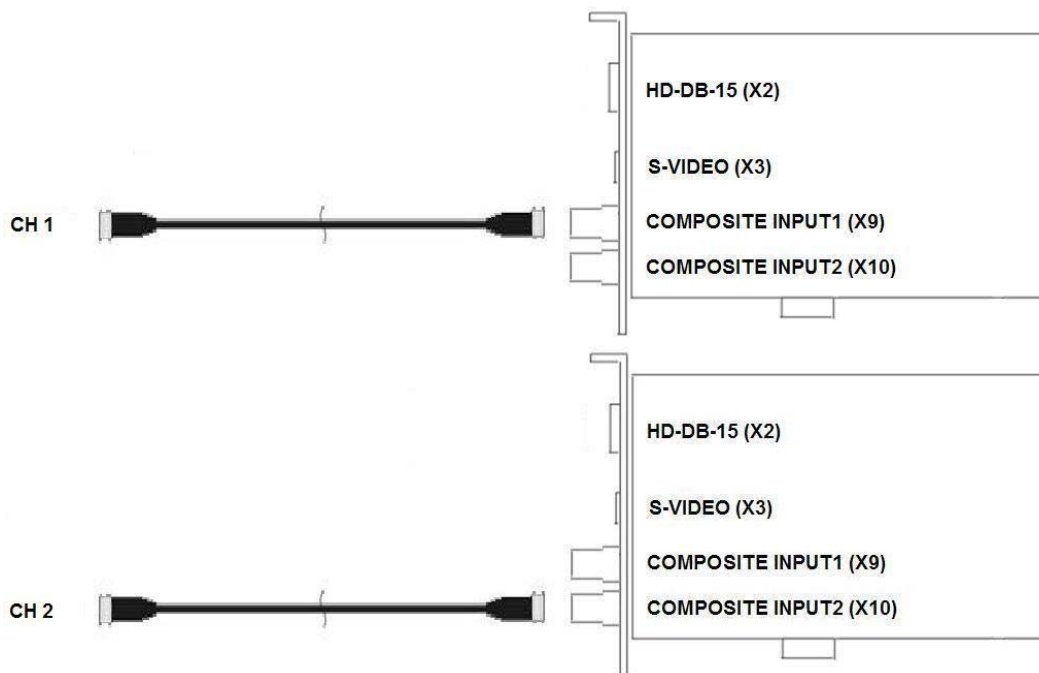


Figure 23: Connectors for the VD-011 Model (part 2)

The following section briefly describes the above depicted cables.

4.7.2 The Video/Power Cable

Figure 24 depicts the connection of a video/power cable to a camera (i.e. PHYTEC VCAM 110-x)

The video/power cable is intended for connection of an S-Video camera. A dual wire with open ends integrated into the cable functions as a power supply for the camera by the framegrabber (red= +12 V, black = Ground).

The HD-DB-15 plug of the cable is connected to the framegrabber's HD-DB-15 connector X2.

Caution:

To avoid short circuits during installation, establish the connection to the camera first before connecting the cable to the framegrabber card! We recommend to establish any connection while the computer is off.

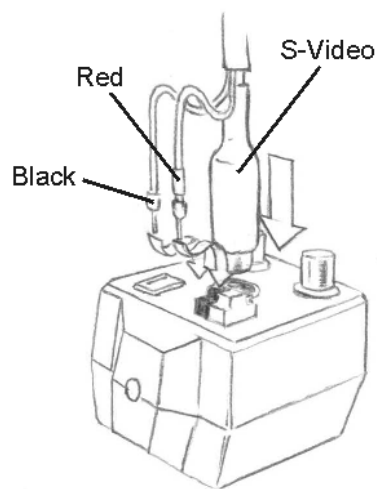


Figure 24: Connecting a Camera (VCAM 110-x) to the Video Power Cable (An Example)

4.7.3 The S-Video Cable

The S-Video cable is connected to the framegrabber using the round Mini DIN socket. The video source (i.e. camera with S-Video output) should have a similar socket.

4.7.4 The Composite Connectors

It is possible to connect the composite outputs (BNC plug) with a video source using a BNC plug.

Note:

If the composite sources contain a RCA connector a RCA/BNC adapter (75 Ω) must be used.

The end that contains the HD-DB15 socket is inserted into the framegrabber.

It is possible to supply up to 4 composite sources and a power supply (pin 14 on the framegrabber socket ②).

The pciGrabber-4express is equipped with two BNC-sockets so it is possible to supply each with a composite source.

In order to digitize an image, the correct channel must be selected in the user's software and in the demo program. It is possible for the included software to automatically recognize which channel is supplied with a signal (*see section 6*).

! Now please proceed to *chapter Fehler! Verweisquelle konnte nicht gefunden werden*. to install the driver software for Microsoft Windows and to start up the framegrabber with the demo application.

5 Installing the Driver

- After the installation of the framegrabber card, please connect the computer to the power supply and turn it on. During start-up the computer's BIOS should automatically recognize the card.

Depending on the operating system, there are two possible scenarios:

1. Either the operating system recognizes the card and searches for the driver or
2. the operating system does not automatically recognize the card (i.e. Windows NT) and the driver has to be installed manually.

Depending on the operating system installed on the computer, please follow the instructions given below to install the driver:

- **Windows 98/ME/2000/XP/VISTA™ :**

After the computer has recognized the card, you will be asked to install the driver.

Place the **PHYTEC Vision Utilities** CD into the CD-ROM drive. Select the „*Search for a better driver*“ option from the *Hardware Assistant* window, and select *OK* to confirm.

The next dialog box that will appear, will give you a list of the drives of the computer (floppy, CD-ROM, and so on). Choose the installation media type / the driver's location (usually the CD-ROM drive) and click *Next*.

Change the path to ***pciGrab4\driver\win2K_98***. Confirm by selecting *OK*.

A list appears naming the drivers found on the CD. Select ***PHYTEC PCI-Grabber*** from the list.

The CD will automatically install the driver onto the computer.

Now the driver has been successfully installed.

Please refer now to *section 5.1*, if you like to install additional drivers.

Then refer to *section 6* to find information on how to install the demo application.

- **Windows‘ NT4.0™ (with Service Pack 6):**

WindowsNT does not automatically recognize the card, therefore the driver must be installed manually. Place the **PHYTEC Vision Utilities** CD into the CD-ROM drive. From the main folder of the CD, select the program *Start.exe*.

The installation software screen will appear.

Select *PCI-Grabber*, and then *Install drivers - WindowsNT4.0* from the menus.

After following the directions from the installation program, the necessary device drivers will automatically be installed. In the window that will appear, confirm a Restart of the computer.

Now the computer should start-up the operating system again.

The driver has now been successfully installed.

Please *refer now to section 5.1*, if you like to install additional drivers. Then *refer to section 6* to find information on how to install the demo application.

- **Windows‘ 95™:**

After the operating system has recognized the card, a window appears, *New Hardware Found*, offering the user to install the driver. From this window, select the „*different position*“ option and confirm with *OK*.

A new window will appear entitled „*select different position*“. Place the **PHYTEC Vision Utilities** CD into the CD-ROM. Choose *search* and in the window that will appear, select the CD-ROM drive.

Change the path to *pciGrab4\driver\win95_98*. Confirm by selecting *OK*.

A list appears on the CD, which names the drivers found. Select *PHYTEC PCI-Grabber* from the list. Now the driver should be automatically installed from the CD to the computer.

In the window that will appear next, confirm a Restart of the computer.

After the start-up of the computer the driver has been successfully installed.

Please *refer now to section 5.1*, if you like to install additional drivers. Then *refer to section 6* to find information on how to install the demo software.

5.1 Additional Drivers (optional)

It is possible to install additional drivers from the CD-ROM, although these drivers are not necessary for the functioning of the card described in this manual.

The **Twain driver** is a standard driver intended for use with graphics-, photo-, and scanner software. The Twain driver reads images and imports the camera's images directly into the application. Thus, the twain driver enables the framegrabber and camera to function like a scanner device.

For additional information on the Twain driver, please refer to the User's Manual on the graphics program that is being used.

If installation of the driver is desired:

Place the **PHYTEC Vision Utilities** CD into the CD-ROM drive and start the file *start.exe*. This file can be found in the main folder of the CD.

The installation software screen will appear.
Select *PCI-Grabber* from the menu. The choose:

- ***Install Twain Driver***

Choose this entry and press "Next". Now please follow the instructions.

6 Start-Up the Framegrabber with the Demo Application

6.1 Installing the Demo Application

With a connected camera, the demo program allows the user to test the card, modify image parameters, and execute simple image operations.

Installation of the software:

- Place the **PHYTEC Vision Utilities** CD into the CD-ROM drive.
- The CD-ROM drive must be selected and the program *start.exe* (found in the CD's main folder) must be started.
- Select the *pciGrabber* from the installation menu that will appear (see Figure 25).
- Click on *install demo*.

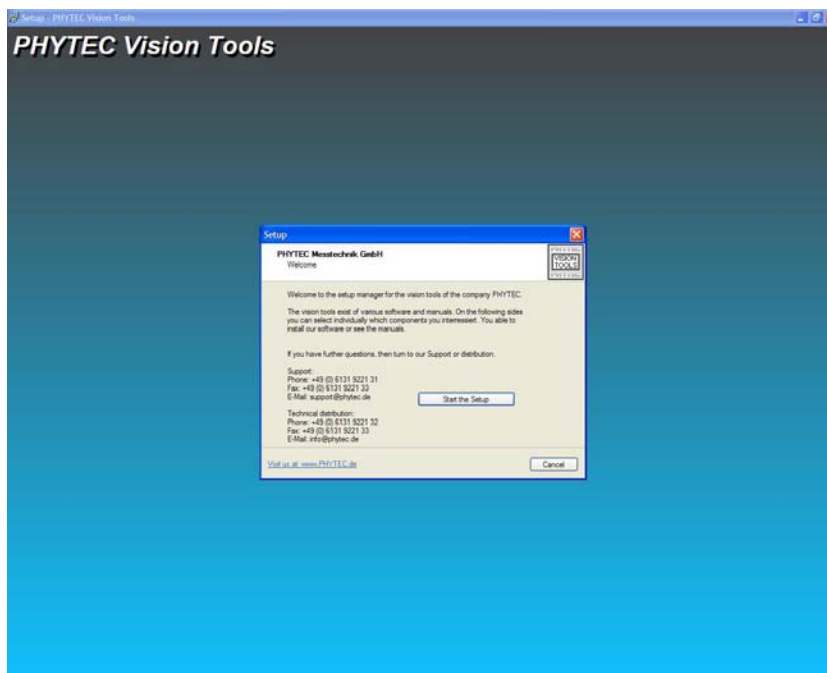


Figure 25: PHYTEC Installation Menu

- Follow the installation instructions and the demo application will be automatically installed on the computer.

6.2 Description of the Demo Software

In order to continue with this section, the demo software and the framegrabber's software driver must be installed properly (*see section Fehler! Verweisquelle konnte nicht gefunden werden.*).

The demo application can be found in the Applications folder of your operation system. The subfolders are named *Phytec / pciGrabber4plus*. Start the *Grab4PCI* application. An empty application window will appear with the menu options (*see Figure 26*).

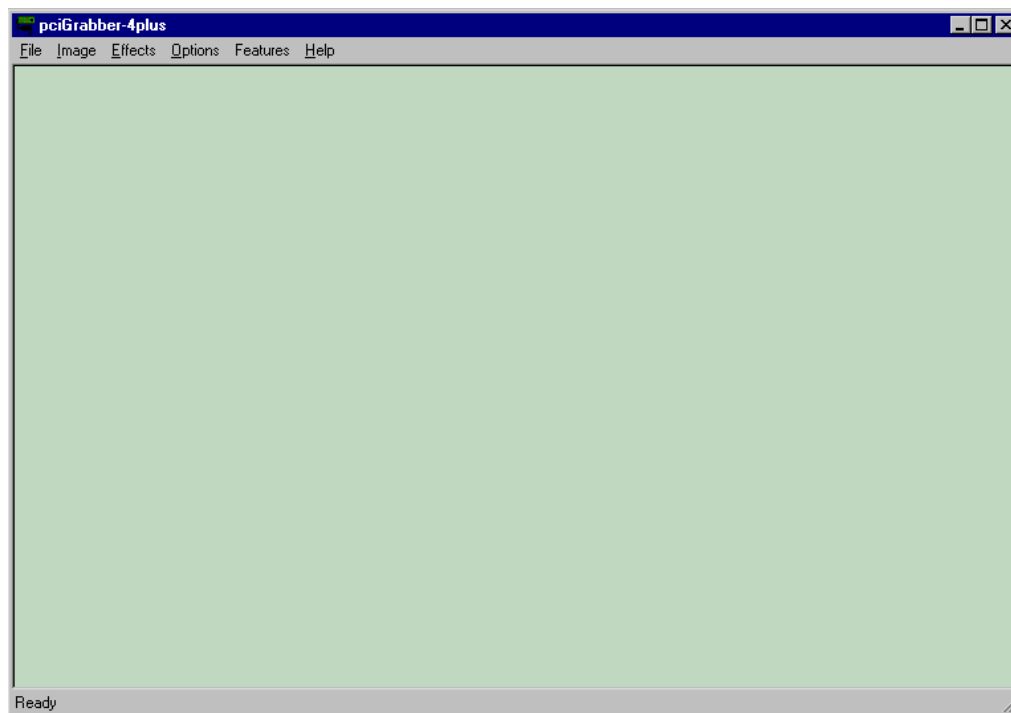


Figure 26: Overview of the Demo Application

Our first task is to start the framegrabber and to display a live image from the camera in the application. Please ensure that a video camera, or another source is connected to the framegrabber and that an image signal is being transmitted.

Click on the *Image* button and the following pull-down menu will appear (*see below*).

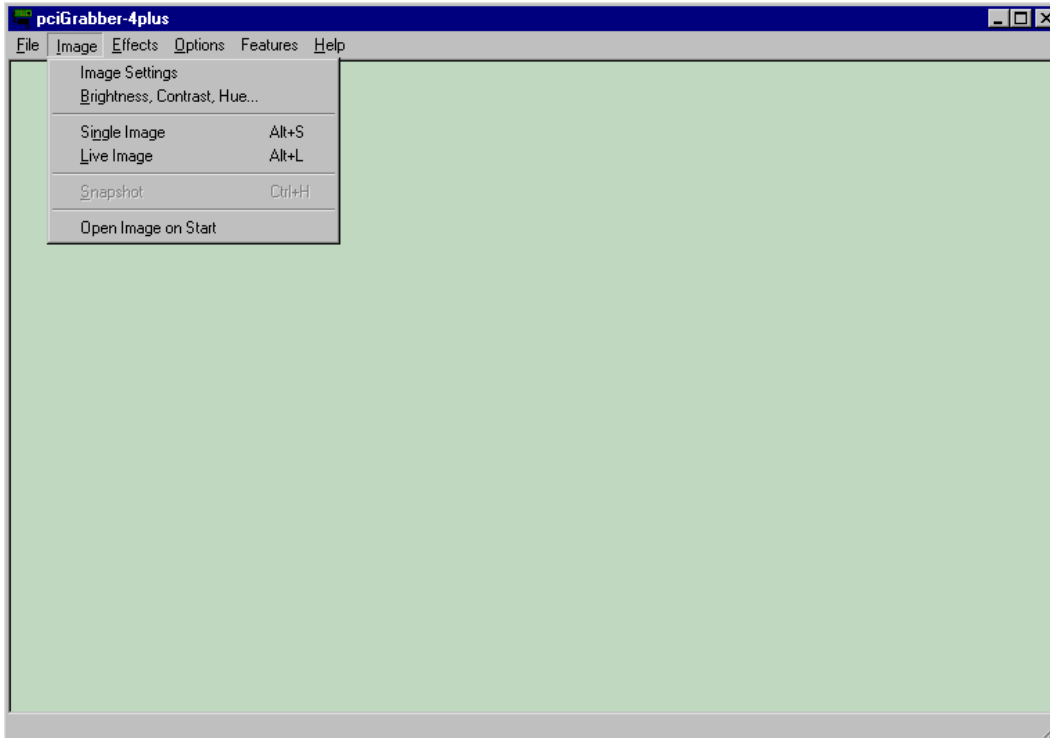


Figure 27: Menu Option: Image

In order to configure the parameters of the image to be grabbed, select the *Image Settings* command from the pull-down menu (*see Figure 28*).

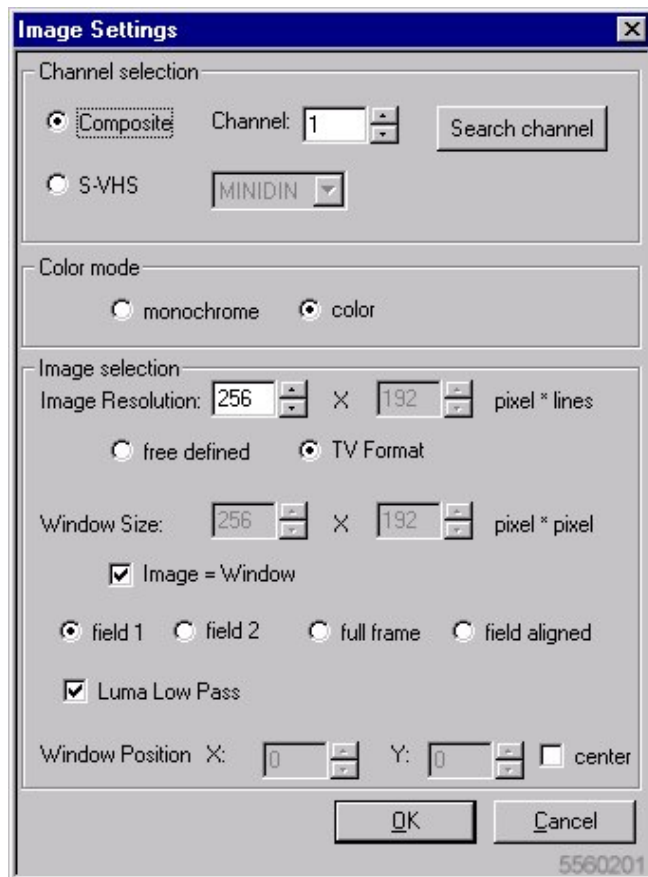


Figure 28: Configuring the Image Parameters

Detailed descriptions of each parameter will be given later on in this manual. In order to test the framegrabber, the live image should be displayed on the computer's monitor. To display the image on the monitor, the following requirements must be met:

It is important to select the proper video input for the framegrabber. In the *Channel selection* field, fill in the type of video source (Composite/ S-VHS) and the input channel that is being used.

The input channels can either be manually entered, or automatically searched for. In order to use the automatic search, click on the *Search channel* button. The first channel with an active video signal that is found is used.

Note:

The automatic channel search does not properly recognize an S-Video source connected to the COMBI socket (socket ②). In this case, the user must manually activate the S-VHS (COMBI) button. If the button is not activated, the image will not appear in color.

In the *Color Mode* field, the user can choose to display the image in color, or in monochrome.

The remaining entries under *Image selection* can retain their pre-configured values.

Exit the menu by clicking *OK*.

Now select the *Live Image* command from the *Image* pull-down menu.

A live image from the selected video source will now be displayed in a new window (see *Figure 29*)

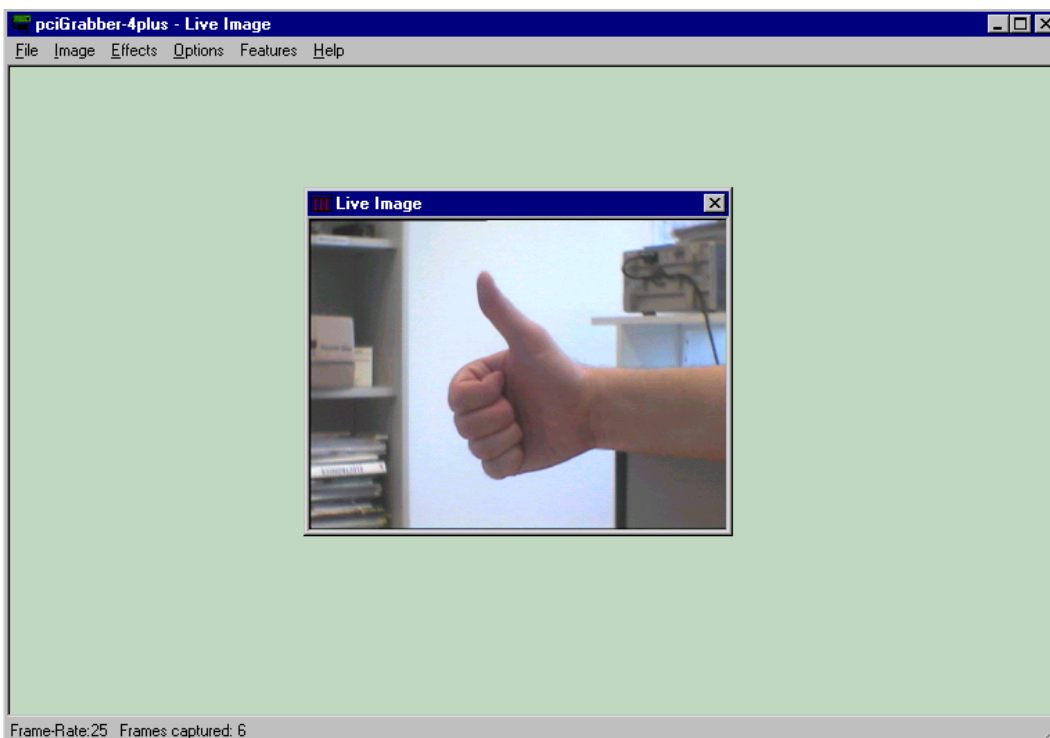


Figure 29: Live Image from the Video Source

If a blue screen appears, this is an indicator that no proper input signal is present on the selected channel. Please examine all cables to ensure that they are properly connected. Also ensure that the camera is receiving power.

If the connections are established properly and the camera is powered, perhaps an incorrect input channel or framegrabber (if several cards are installed) was selected.

Additional help on common problems are described in the appendix.

Note:

When operating multiple framegrabber cards in a computer, please select at first the grabber card you like to be the active framegrabber for this application.

Designating a grabber can be done in the *Options* menu.

The *Frame Rate* display can be found on the status bar of the main window. The value represents the number of images that are generated per second in the live video window. The value is dependant on the size of the image, and the capacity of the computer, because the digitized image must be transmitted from the computer's RAM to the graphics adapter to show up on the screen.

Note:

Despite the processor's capacity, the framegrabber always stores image data in real time in the main memory (RAM) of the PC.

Further processing of the data only is dependant on the CPU of the computer.

The status bar further contains a counter that displays the total number of live images captured (*Frames Captured*).

When the counter has reached 255, it automatically begins a new sequence starting with 0.

This information can also be used to indicate whether the framegrabber is active or not.

6.3 Demo Software - Detailed Description

This section describes in greater detail the functionality of the demo software as well as the features of the menus.

The *Image Settings* menu (see *Figure 30*) contains parameters that influence image generation and depiction:

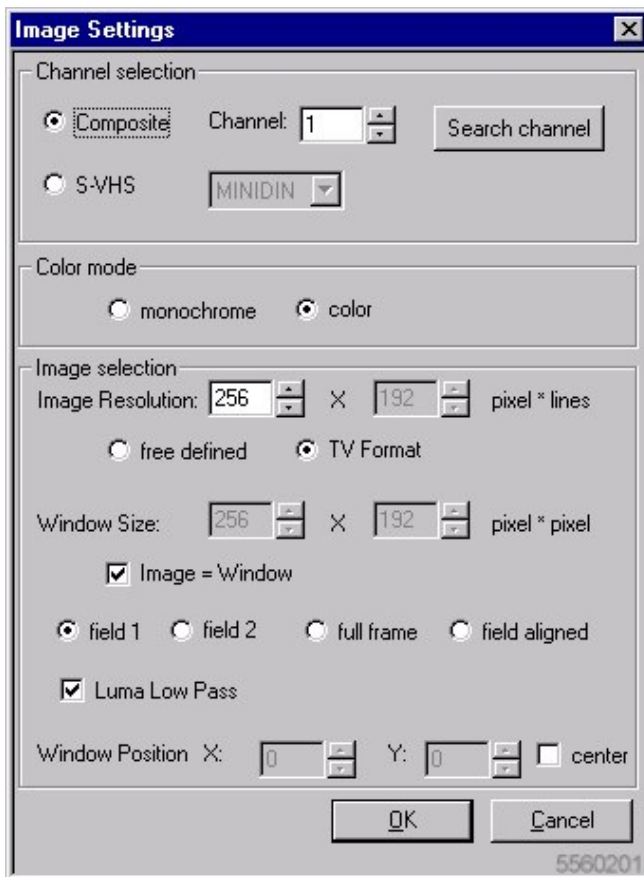


Figure 30: „Image Setting“ Menu

The parameters can only be configured before a live image is displayed. During capturing of live images, parameters cannot be configured.

The section entitled *Channel Selection*, offers parameters for video source types and channel selection.

Click on either the *Composite* or the *S-VHS* button to select the appropriate signal type.

- **Composite Sources**

Composite refers to both of the HD-DB15 sockets and the BNC sockets (VD-011).

From the *Channel* menu, select the appropriate input channel for the connected camera.

Clicking on *Search Channel* allows the grabber to search for an active input channel. The program configures the first channel with a video signal.

- **S-Video Source**

S-Video (or „S-VHS“) – Two possibilities exist for connecting a source to the framegrabber. Select the appropriate socket from the dialog box.

MINIDIN – The image source is connected to the round mini DIN socket.

COMBI – The image source is connected to socket ② via the video power cable (S-VHS and power supply).

Note:

Using the video/power cable creates some limitations. The parameter *S-VHS – COMBI* must be manually selected in order to activate a video source at this connection.

The user can choose to display an image in color (when using a color video source) or monochrome by using the *color* and *monochrome* buttons.

“*Image Selection*“, found in the lower section of the window, can be used to configure the size and resolution of the image.

The *Image Resolution* parameter is used to configure the image’s resolution (which can be referred to as some sort of image quality). It specifies the number of pixels that are digitized from the video signal. Since this parameters define the size of the digital image in respect to the resolution of the video signal, this procedure is also called *scaling*.

The parameters divide into x-direction for the pixel number and in y-direction for the line number. Both values can be changed separately if activating the radio button *free defined*.

Please note, that the image will be displayed distorted (stretched or shrunk) if the 4:3 ratio is not adhered to. (This width to height ratio given by the television standards).

The *TV Format* button prevents image distortion by automatically adhering to the 4:3 ratio (width/height relationship). For example, if given the number of pixels, the number of rows is automatically calculated with the 4:3 ratio.

The parameters *Window Size* can be used to extract a section of the image, and display this section only instead of the whole picture on the monitor. This feature is also called *cropping* the image.

This section can be smaller than the original viewing field of the camera. If the entire digitized field is to be displayed, then activate the *Image=Window* checkbox.

The *Window Size* does not distort the image geometry because it is not a scaled section, rather than a cut out section.

Note:

Please note that processing *scaling* and *cropping* is done in real time by the framegrabber. The framegrabber stores the image in the format as it is displayed on the monitor. This is very beneficial because no CPU power is needed for this function.

A brief explanation of analogue television technology will lead to a better understanding of the buttons *field1*, *field2*, *full frame*, and *field aligned*.

A television image (thus a video signal) is made up of two interlaced image portions, called *fields* (see *Figure 31*). These half frames (fields) are generated in sequence and then displayed on a screen (i.e. of a television set).

The interlacing of the images reduces the flickering that can occur with TV images.

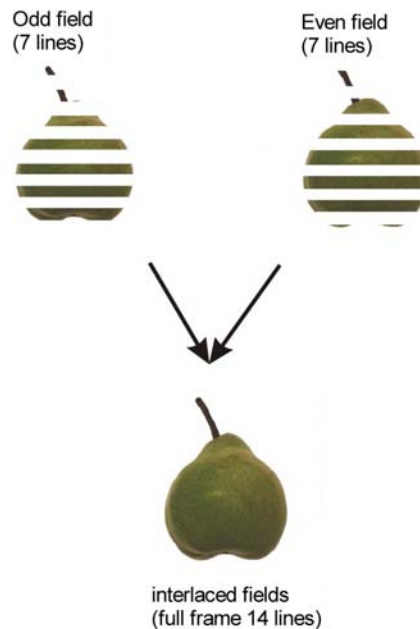


Figure 31: Creating a Full Image: Two Fields, Each with 7 rows

According to the PAL-standard, each signal contains 625 rows. The rows are divided into field frames: the first field (*odd field, field1 with rows 1-625*) and a second field (*even field, field2 with rows 2-624*). An image section is fully recognizable by one of its fields. The image's vertical resolution is reduced to the half, since the image is only represented by 288 rows. (excluding the invisible rows that precede and succeed the image as well as test and data rows.) A total of approximately 576 from 625 rows remain visible

Digitizing a field is time efficient; compare 20 ms for a field image to 40 ms for both fields (*full frame*).

If the same field (i.e. the first) needs to be digitized repeatedly, there is a pause of 20 ms between the processes.

Digitizing a full frame can create a distorted image if the object moves too quickly. The object is in a different location in the first field than it is in the second, creating a comb effect. The image may appear as shown in *Figure 32*).

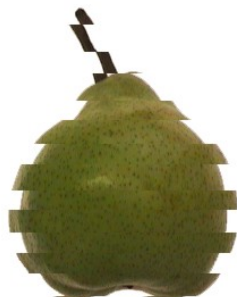


Figure 32: Comb Effect That Occurs with Quick Moving Objects

The parameters described above can be changed in the demo software, so the effects mentioned can be tested.

With vertical resolutions smaller than 288 rows, it is advisable to digitize a field. The *field1* (first, odd field) and *field2* (second, even field) buttons can be used to set the framegrabber to field acquisition mode.

If the number of rows exceeds 288, then both fields must be digitized. To digitize both fields, activate the *full frame* option. If a number larger than 288 is entered into *Image Resolution*, the *full frame* mode is automatically selected.

The *field aligned* mode allows to double the number of digitized fields per second. This eliminates the 20 ms pause between the acquisition of similar fields.

If simply even and odd fields would be consecutively put to exactly the same space on the display, the visual impression would be the image contents shifting up and down by half a line.

This occurs because the two fields had not been *interlaced* to form a full frame. When activating the *field aligned* mode, the framegrabber automatically shifts the second field vertically for one half row, so that the second field can properly fit with the first half frame, creating an image with field resolution every 20 ms.

This configuration of *field aligned* is thus helpful when the user likes to digitize images consecutively with a maximum of 288 rows, at a high framerate (1 field in 20 ms).

If the horizontal resolution is set to less than 360 pixels, the checkbox *Luma Low Pass* should be activated.

This low pass filter will smoothen the image according to the lower resolution. The low pass filter will be activated automatically if the horizontal resolution is less than 360 Pixel and deactivated otherwise.

The parameter *Window Position* can be used to determine the position of the image section defined by the cropping parameters within the total image. The values represent the position of the upper-left corner of the window. In order to center the cropped image portion in the original image, check the box *center*.

Changing the *Window Position* parameters will move the cropped image within the entire image. This is referred to as *panning*. Note that panning can only take in effect if the *Window Size* is less than the image size defined by the parameter *Image Resolution*).

6.4 Image Control

During the capturing of a live image, the image control dialog can be opened by selecting *Image - Brightness, Contrast, Hue*. The dialog box shown in Figure 33 will appear. Changing the slider controls will affect the values of brightness, contrast, color saturation and hue. The changes are immediately applied to the framegrabber, so that the corresponding effects can be seen in the live image.

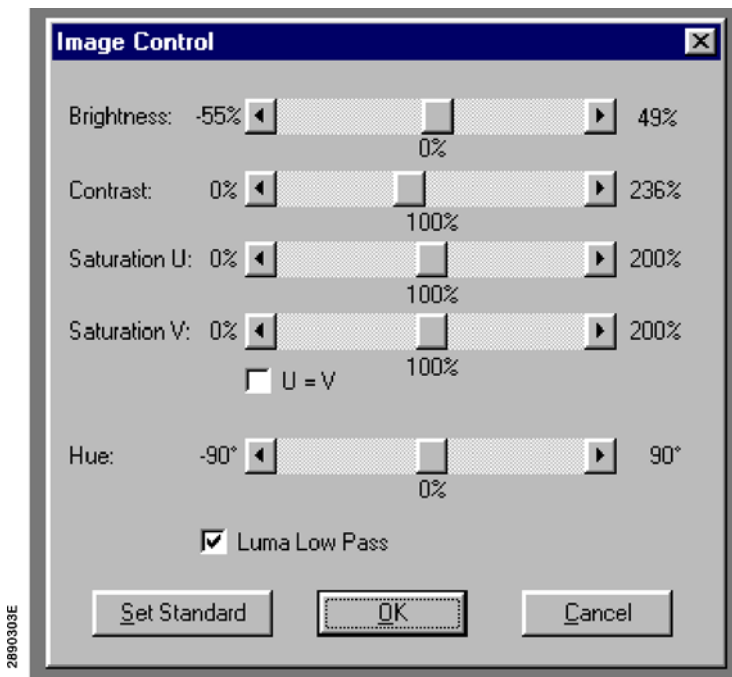


Figure 33: The Image Control Window

Use of the slider control *Brightness* allows to adapt the brightness of the image. The control *Contrast* changes the contrast of the image.

For the adjustment of the color saturation two controls are available: *Saturation U* and *Saturation V*. This allows separate manipulation of the saturation in the red- and blue/violet region. With the control box $U=V$ both controls can be locked. In this way you are able to change the color saturation without changing the color tone.

The *hue* control is only applicable on NTSC-system video sources. This control serves for the correction of the color tone, in case a phase shift has occurred during transmission. Those interference's can only be present in NTSC systems. The PAL system corrects color tone failures automatically, so that the hue control has no effect. Please leave the slider control in the center position in this case.

Note:

Modification of the hue (i.e. white balance) can also be done on both NTSC and PAL systems by moving the saturation sliders separately (in general, it is better to adjust the white balance at the camera or video source if possible).

6.5 Additional Functions of the *Image* Dialog

- Using the *Single Image* feature, a snapshot is taken and displayed on the screen. In this mode, the framegrabber only performs one single capture.
- The parameter *Image Settings* defines the image parameters.
- Using the parameter *Live Image*, a live image is captured and shown on the monitor. *Image settings* also defines the image parameters in this mode.
- Snapshots can also be taken during live image acquisition using the *Snapshot* option.
- The snapshot will be displayed in a new window. Multiple snapshots can be taken.
- Snapshot-Windows that appear on the screen are automatically numbered.
- Checking the option *Open Image on Start* allows to save the parameters configured in *Image Settings* dialog and will automatically open a live image window with these parameters upon each start of the application.
- Adding the demo application to the auto start group enables the computer to display a live image after start-up, without any intervention by the user. The image will be captured with the parameters defined and saved before.

6.6 Crosshair function (Overlay)

Several types of crosshairs can be overlaid on the live image. This can be useful for example to center an object in the middle of the image.

The parameters for this function can be found under the menu option *Effects*. All of the crosshairs, or a combination of them, can be overlaid on the image.

6.7 Basic Parameters

The basic parameters of the framegrabber can be set in the *Options* menu.

Basic Settings contains the following menu:

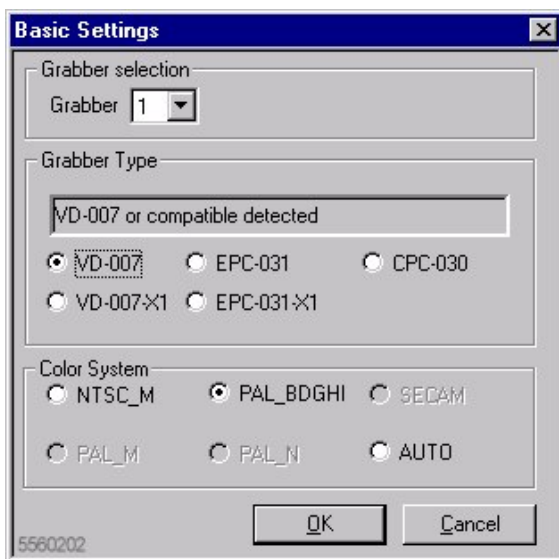


Figure 34: Basic Settings Menu

When operating multiple framegrabber cards in the computer, the demo application has to be logically connected to a specific framegrabber card. This is done by selecting the appropriate card number in the *Grabber selection* field. If there is only one framegrabber installed in the computer, then the framegrabber is automatically activated and assigned with the number 1.

Grabber Type shows information about the model of the framegrabber that is installed in the computer and currently activated for this demo application (see *Grabber selection*).

VD-009, VD-009-RS6, VD-009-X1 or VD-009-X1-RS6 (depending on framegrabber model) denotes the *pciGrabber-4plus*.

VD-011 or VD-011-RS6 denotes the *pciGrabber-4express*.

Note:

When using an older PHYTEC framegrabber model, the model is denoted as „VD-007 or compatible“. In this case, the exact type of card cannot be recognized and model VD-007 is automatically configured. In this case, select the installed framegrabber from the list manually.

Color System configures the color system to be used with the framegrabber.

PAL is mainly used in Europe and NTSC is used in the USA.

While capturing live images, these parameters cannot be changed.

Addition Settings and *Type Casting Settings* are described with *Add Live Images* and *Arithmetics* later on in this manual. All of these features can be found in the *Features* menu.

6.8 Special Functions

The demo software offers several basic functions to manipulate and analyze images.

- **Display Histograms**

Histogram enables a histogram to be calculated from a static image, i.e. an image obtained using the *Snapshot* option.

A histogram provides the frequency distribution of the grey- and/or color values of an image.

The relative frequency of the corresponding intensity values are represented by brightness (intensity) of the individual color channels (see Figure 35).

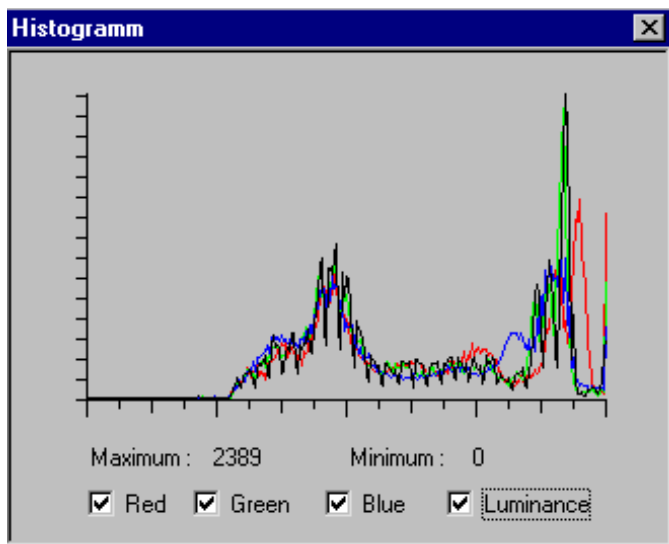


Figure 35: *Histogram*

The X-axis depicts the value range between 0 and 255. Using the check boxes in the histogram window, the distribution of the intensity of the individual color channels or the luminance channel can be turned on/off separately.

Note:

A histogram can only be created from a still image, and not from a live image. To create a still image from a live image, you can use the snapshot function.

- **Analyzing Colors**

Select the *Color Meter* option to open the window shown in *Figure 36*.

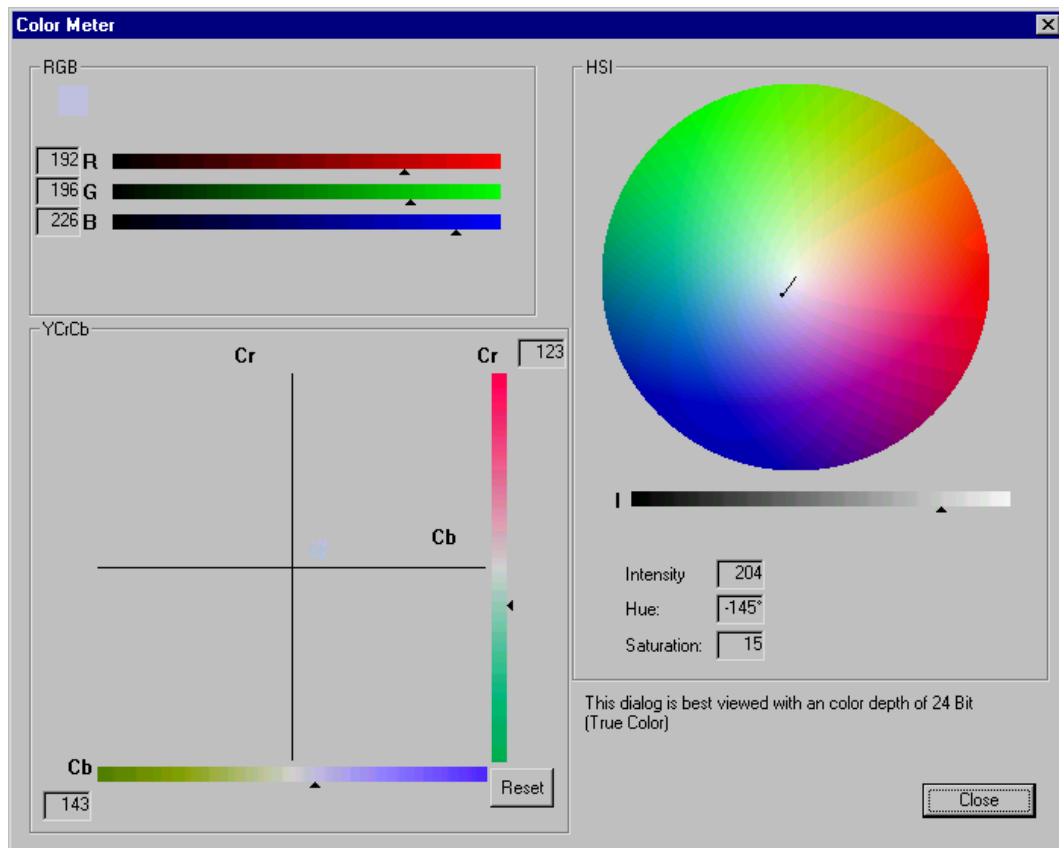


Figure 36: Color Meter

The color meter option is available only during live image capture. The color meter displays various color models for the color values of pixel exactly in the center of the image.

A small crosshair that appears in live image indicates the center of the image.

The RGB model displays the color values for red, green and blue as a pointers on the intensity bars and numerical values.

The YCrCb model show color values as color bars and plots them on a coordinate system.

Thus, color information can be observed over an extended period of time.

The *Reset* button erases the existing coordinate graph and creates a new graph.

The HSI model displays the color values in a chromatic circle. The length of the vector indicates the saturation level, and the direction of the vector indicates the hue. Brightness is displayed in a gauge at the bottom of the window.

Each value is also output in numerical format.

- **Color Bars (Test Pattern)**

Select the *Color Bars* option in order to test the framegrabber.

The color bars are generated by hardware and not by the demo software. The number of vertical bars displayed depends on size of the image. All color bars are displayed with a horizontal resolution of approx. 515 pixels.

- **Arithmetic Operations on Static Images**

The *Arithmetics* menu option provides some simple arithmetic operations on snapshots (*see Figure 37*).

For example, images can be added, subtracted, multiplied or divided pixel by pixel. In addition, a constant can be added to each pixel (brightness changes) or the constant can be multiplied with each pixel (contrast change).

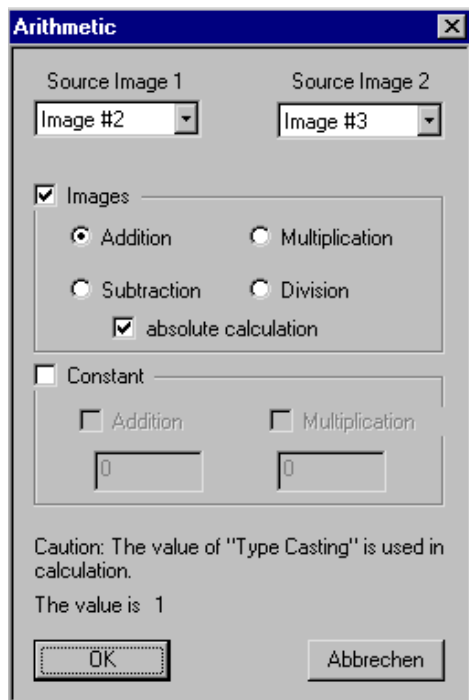


Figure 37: Arithmetics Menu

Images to be manipulated can be selected from the *Source Image 1* and *Source Image 2* boxes. The number behind *Image#* corresponds to the number of the image window.

Arithmetic operations can be selected from the *Images* entry. The user can also choose to perform an absolute calculation.

When performing an absolute calculation, negative values are not allowed. Eventually these negative values will lead to a meaningless and incorrect result.

The *Constant* option allows a constant to be added to each pixel (change of brightness) or to be multiplied with each pixel (change of contrast).

All arithmetic operations can be normalized. This is important if the result is expected to be outside of the displayed range of values. (Each color channel has a range from 0 to 255). On principle, values greater than 255 are set to 255, and pixels with values less than 0 are set to 0. Thus, resulting values are probably cut.

The use of normalizing prevents, for example, the creation of white images caused by multiplying pixels.

The normalization factor can be selected from *Options* menu in the *Type Casting* dialog (see *Figure 38*).

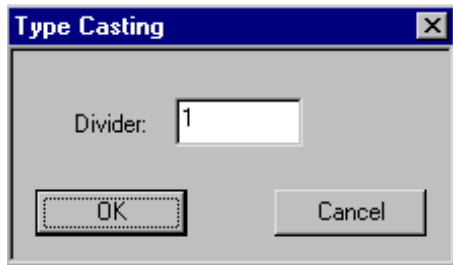


Figure 38: Selecting the Normalization Factor

The current value is displayed in the bottom section of the *Arithmetic* menu.

Caution:

Incorrect settings of the normalization factor will provide bad results with arithmetic operations (i.e black or white images).

- **Add Live Images**

The *Add Live Images* option allows to compile up to 1000 consecutive live images into a single image.

The desired number of images can be selected under *Options / Addition Settings* (see *Figure 39*).

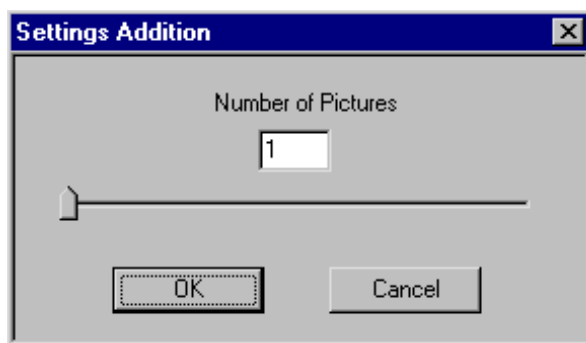


Figure 39: Number of Images

This feature can also be used to reduce noise levels when capturing images or to reduce the appearance of moving objects in comparison to the background.

After the addition process the resulting picture is normalized so that the original brightness is retained.

The length of the process depends on the number of images that were added and the capabilities of the computer.

The operation's status is displayed as a percentage in the lower section of the window.

Caution:

In order to ensure that the brightness for added images has the same level as single images, the parameters for the normalization factor (*Type Casting*) change simultaneously with the number of images (*Addition Settings*).

The normalization factor must eventually be re-configured when using additional arithmetic functions.

- **I/O Port Test**

Select the *I/O Test* command from the *Test Hardware* pull-down menu. The window shown in *Figure 40* will appear.

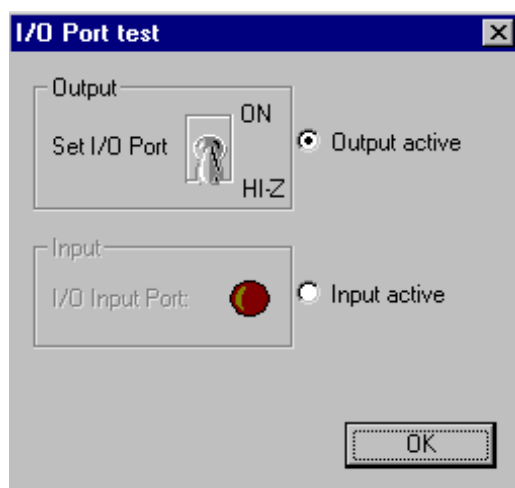


Figure 40: I/O Test Menu

This dialog box enables the user to test the I/O port. The port is an input / output switch that is controlled by a transistor on the framegrabber card.

The port can be connected via the framegrabber's HD-DB15 plug (2) (pin 9 = signal, pin 10 = ground).

If *Output active* is selected, the port pin acts as an output. The *Set I/O Port* switch can be used to activate the output pin (ON = connection established to ground) or deactivate the output (HI-Z = high impedance).

Selecting *Input active* allows the user to test the *I/O Input Port* function. The simulated red light indicates if the port reacts to an external signal.

If the pin remains open (unconnected), the input is activated after the button has been selected. The red light will illuminate and indicates that state.

- **Option Port Test**

The *Option Port* feature is located in *Features/Test Hardware*.

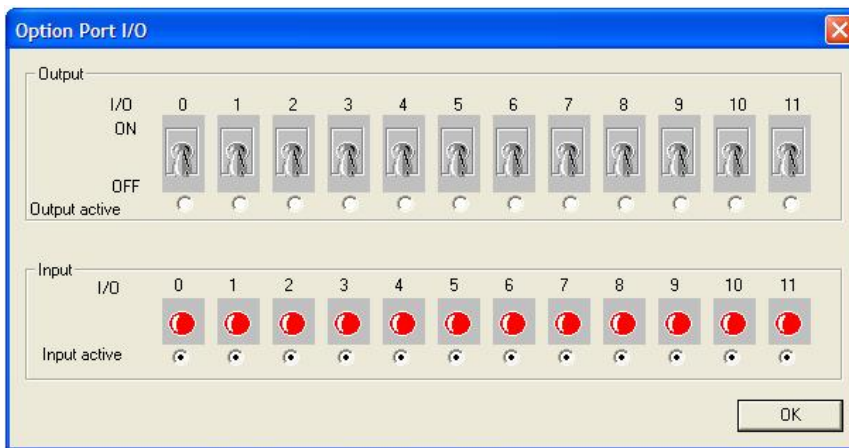


Figure 41: Option Port Menü

This feature permits to control the I/Os on the Option Port (X6). It is possible to turn the I/Os on or off. Further it is possible to configure the I/Os individually as inputs and read back the pin's state.

- **DIP Switch Test:**

The *DIP switches* test feature is located in *Features/Test Hardware*.

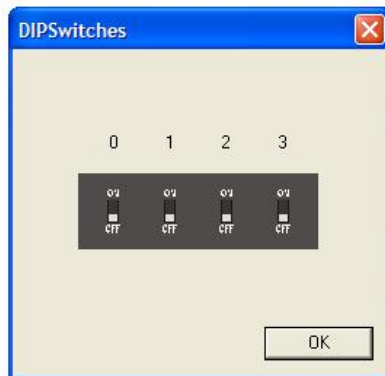


Figure 42: *DIP switches Menu*

This function permits to read the DIP switch settings.

Attention!

Only RS6 variants contain this feature.

(VD-009-RS6, VD-009-X1-RS6 and VD-011-RS6)

- **Relay Test**

The *Relais* test feature is located in *Features/Test Hardware*..



Figure 43: *Relays Menu*

This option permits to control the relays. It is possible to turn the relays on or off.

Attention!

Only RS6 variants contain this feature.

(VD-009-RS6, VD-009-X1-RS6 and VD-011-RS6)

6.9 Storing Images, Ending the Program

The menu option *File* enables users to store live images (as snapshots), still images and arithmetically processed images. The options *Save* or *Save as* allow the images to be saved with an index number given by the program, or with a name given by the user.

The images are saved in bitmap (*.BMP) format and can be viewed and processed with any graphics software.

The *Close* option consecutively closes static images as well as the live window.

Exit closes and leaves the program.

6.10 Getting Started with Linux

Using the PHYTEC framegrabber cards ,pciGrabber-4plus‘, ,pciGrabber-4express‘ and ,eGrabber-4plus‘ with Linux is easy, because these cards are supported by the standard framegrabber-driver ,bttv‘ (Video-4-Linux, V4L or V4L2)

The pciGrabber-4plus/express has an own card definition in the bttv-driver. If your bttv-driver version does not contain this card definition, please upgrade to a newer bttv version.

The PHYTEC-cards are included in version 0.7.107 or higer. Current versions can be found on the bttv-driver page www.bytesex.org.

Please note that the bttv-driver does not detect the pciGrabber-4plus/express automatically.

The driver has to be configured by the following procedure:

Step 1: Choose the card number which corresponds to your grabber version from the list below.

Step 2: Edit the *file etc/modules.conf* using a text editor. Enter the card number as show below:

```
/etc/modules.conf:
```

```
...
alias char-major-81 videodev
alias char-major-81-0 bttv
options bttv card=106 ← enter your card number here
...
```

List of card numbers		
Card No.	framegrabber Model	S-Video-Input
106	VD-009-X1	Mini-DIN-plug
107	VD-009-X1	Combi-connector
108	VD-009	Mini-DIN-plug
109	VD-009	Combi-connector
106	VD-011	Mini-DIN-Buchse

Hints:

- At the moment, VD-011 uses the same card number as the VD-009-X1, because they are compatible. The VD-011 appears because of that as a VD-009-X1 in the Linux system. Please read the FAQs under www.PHYTEC.de for latest information.
- The selection of the card number also defines, whether the Mini-DIN or the Combi-Connector is used as the video-input. In difference to the driver for Microsoft Windows, this selection cannot be done during runtime but only by the card number selection. This is, because the bttv-driver does not support more than one s-video-input.
- For testing the correct function of the driver, we suggest to use the XawTV application.
- With the pciGrabber-4express it might occur that the PCI bus allocation for the devices does not fit anymore, because this card adds an additional PCI bus and Linux is not capable to update the bus structure automatically. Thus, the devices must be relocated. The graphics card can be configured with the SaX2 function automatically or manually in Xorg.conf. If further information is needed, please ask your Linux provider.

Part 2

Programmer's Manual

7 Driver Software

This section gives you the information how you can access the *pciGrabber-4plus/express* with your own program.

The driver library provides you with a collection of functions, which are able to configure the framegrabber, which can inquire the status of the framegrabber and start the digitization.

Software drivers for different operating systems are available.

In this manual drivers for

- Windows 95/98/ME/XP/VISTA
- Windows NT 4.0
- Windows 2000
- DOS

are explained.

Note:

In order to obtain the latest information regarding the driver and the availability of additional drivers, please read *readme.txt*. (This file can be found on the installation CD.)

The next section describes the technical features of the framegrabber and explains television standard in greater detail for a better understanding of the framegrabber's functionality.

7.1 Technical Basics

7.1.1 Block Diagram of the pciGrabber-4plus

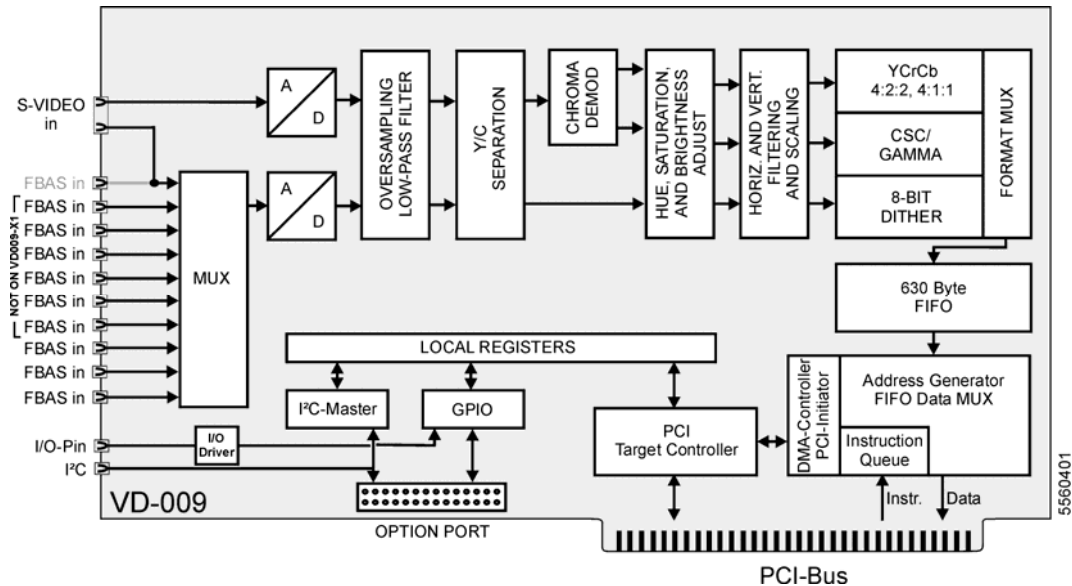


Figure 44: Block diagram

Figure 44 shows the block diagram of the pciGrabber-4plus. The composite input signal is connected to a 9:1-video multiplexer (VD-009-X1 only), which is controlled by software. The following A/D-converter digitizes this signal. All image sources which provide a color video signal corresponding to the CCIR- standard „PAL (B,D,G,H,I)“, „NTSC (M)“ can be used.

In europe image sources generally provide PAL-signals. In this manual we assume that PAL-signal sources are used.

Via the S-VIDEO-input luma- and chroma-signal can be supplied separately (for example from a S-Video-camera or S-VHS-videorecorder). For the color component of the signal a separate A/D-converter is used, which improves the quality of the image.

Also monochrome (“black/white”) videosources can be connected to the *pciGrabber-4plus*. The processing of grey scale pictures with 256 grey levels is already provided in the framegrabber and can be activated by software. Applying black/white sources, the sharpness of the image can be improved by deactivating the luma notch filter by software.

After the A/D-converters follow operational components, which decompose the data stream of the image into its components: After the chroma-demodulator the data are separated according to brightness (Y) and color portion (Cr/Cb). Subsequently follows the digital correction of brightness, contrast, color saturation and the formatting the image (scaling and cropping).

The following *video format converter* produces the data formats, which are provided by the *pciGrabber-4plus*. Via a datamultiplexer the required format is selected and stored in the 630 Byte FIFO memory. The FIFO is an interface to the following PCI-bus interface, which is responsible for the data transfer through the PCI-bus.

The image data are transferred to the main memory of the PC by DMA. For each field a separate DMA-channel is used. The transfer can be organized in different ways. For this reason a *pixel instruction list* for each field is used, which is denoted *RISC-Program*, for the PCI-controller of the *pciGrabber-4plus*. The principle of the pixel instruction list is explained in detail in Chapter 7.1.2.

Via the PCI-controller the access to the *local registers* is managed. This allows the adjustment of the parameters of the framegrabber and reading back the actual status. This registers are also used to actuate the I/O- lines defined by the user, and to drive the I²C-interface integrated in the framegrabber.

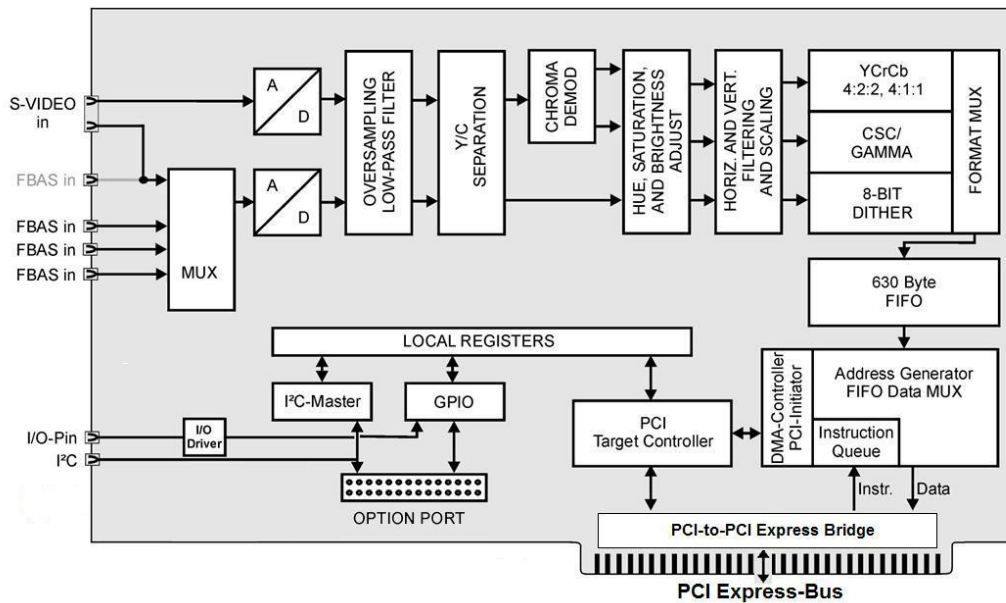


Figure 45: Block diagram

Figure 45 shows the block diagram of the pciGrabber-4express (VD-011). In addition to the pciGrabber-4plus (VD-009-X1), this card features an additional PCI-to-PCI Express-Bridge to adapt it to the PCI Express bus.

Furthermore, the video input connectors differ from the pciGrabber-4plus (not shown in the diagram).

7.1.2 The Video Signal and Capturing Process

The standard video signal, which is processed by the framegrabber, contains 625 lines, which are divided into two fields (see Figure 46). The first field (odd field) contains lines 1 to 313, the second (even field) the lines 314 to 625. The fields are interlaced, in order to reduce flicker of the TV-picture. From a spatial view, line 314 follows line 1. If a full screen is to be displayed, the two consecutively received fields must be interlaced as shown in *Figure 46* (within a TV set, this is achieved by electrical circuitry, in the PC's graphic memory by software).

Besides various retrace- and blanking lines, the video signal contains lines for control and data purposes and lines for teletext information, which restricts the actual image size to two fields of 288 lines.

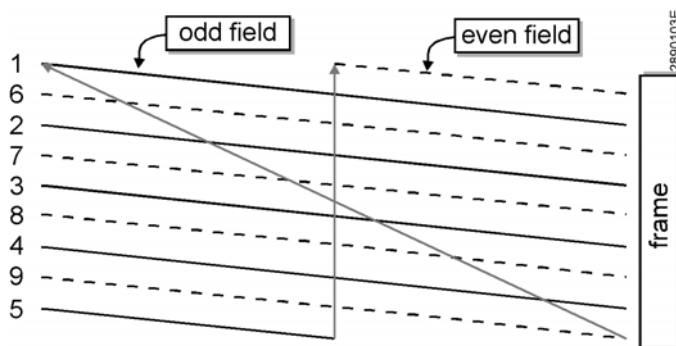


Figure 46: Interlaced image (Example with 9 Lines)

Each field is created within 20ms. One field provides already the complete image, but the vertical resolution is reduced by two. For many applications this might be sufficient, so that after 20 ms a digitized image is already available. In case the resolution in X-direction can also be reduced, we can obtain an image without distortion. However a reduction of the resolution in X-direction can not speed up the digitization process, since the time base is fixed.

If full TV-resolution is required, time has to elapse until both fields are digitized (40 ms). Both fields follow one after another.

In order to allow the interlacing of both fields, the last line of the odd (the first) field, is reduced to the half. Thus, the first line of the second field also contains only a half line.

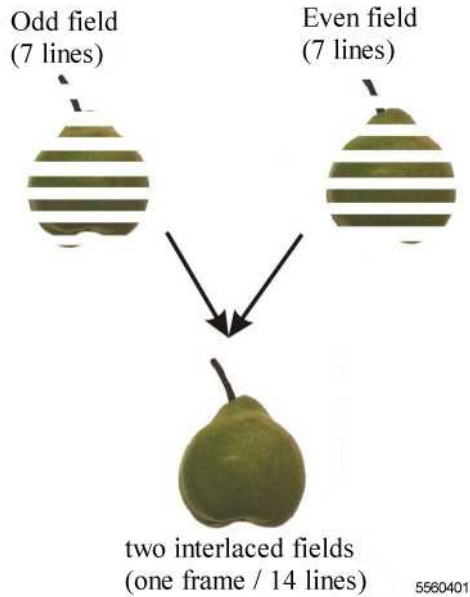


Figure 47: *Fields and Frames*

For fast moving objects it might happen that the object moves noticeably between the capture of the first and second field. Thus, both fields do not match anymore, which will cause horizontal blurring (comb effect). For this reason, quite often only one field is used with a reduced resolution.



Figure 48: *Moving Objects Cause Comb Effects*

7.1.3 Transfer and storage of color

Color and brightness are always separated for the transmission by the TV-systems. Transmitted are the brightness (luma signal, Y-signal) and the color differential signal (chroma signal). This signal defines the color of a pixel by the hue and color saturation.

The TV standards reduce the bandwidth of the color signal in comparison to the brightness signal. This means the color information of a picture is more ‘blurred’ than its corresponding brightness information. This might be compared to a painter, who at first makes a sketch with a sharp pencil and then colors the areas with a broad brush.

The Y-bandwidth of the PAL (B,G,H,I) - system is 5 MHz, and the bandwidth of the chroma signal is 1.5 MHz.

The chroma signal is also denoted as U/V signal for PAL standard or Q/I-signal for the NTSC standard. V- and I-signal define the reddish colors, whereas the U- and Q-signal define the bluish/violet colors. Both signals together are denoted the *Cr/Cb signal* (chroma red / chroma blue).

With the triplet (Y,Cr,Cb) the brightness and color of a pixel are completely defined. These values are ready to be used without further evaluation for image processing in respect to color recognition or color control.

Frequently the definition of a pixel is preferred in the red, green and blue (R,G,B) notation. The transformation according to CCIR-recommendation for PAL is achieved with the following matrix:

$$\begin{pmatrix} R \\ G \\ B \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1,371 & -191,45 \\ 1 & -0,338 & -0,698 & 116,56 \\ 1 & 1,732 & 0 & -237,75 \end{pmatrix} \cdot \begin{pmatrix} Y \\ U \\ V \\ 1 \end{pmatrix}$$

The *pciGrabber-4plus/express* is able to convert the images into the RGB-format and stores the RGB- color triplets in the memory. This format provides a good base for further processing.

Often the YCrCb-format is more suitable for storage or transfer of image data, since the data volume is less. Instead of three Byte only two Byte (one word) are required. The lower eight bits contain the brightness and the eight upper bits specify the color portion (Cr/Cb).

For each Y-value alternately only one color portion Cr or Cb is associated. So each pixel has only one part of color information either the red or blue portion. The missing information can be obtained from the neighboring pixel. The color is transferred and stored only at the half resolution of the brightness. Since the bandwidth of the color information is already reduced by the TV system, this procedure does not mean a real loss. This data format is denoted as YCrCb 4:2:2.

The first pixel of each line delivers $Y_1, Cr_{1/2}$, the second $Y_2, Cb_{1/2}$ etc.

Caution:

For the correct recognition of the color information of an image, four subsequent fields are required to be digitized. Therefore it is not sufficient to connect the video source only for a short period or to connect the video source only for the duration of the digitization of one field.

In addition the recognition of a field might not work correctly at the beginning, in case another not synchronized signal from a different camera is applied. In case of fast switching between two signal sources the digitized image might be incorrect and it is recommended to observe some time delay.

7.1.4 Data storage by DMA and RISC-Program

This section describes the transfer of the data to the main memory and the storage of the pixels to the addresses specified by the user.

As mentioned before, the transfer of the data is accomplished via two DMA-channels, one for the odd and one for the even field. During the time of digitization the DMA-controller of the *pciGrabber-4plus* is controlling the PCI-Bus and is *master*. The data are transferred in real time along the PCI-bus to the main memory. This is possible because of the high transfer rate of the PCI-bus.

Delays of the data transfer or time intervals the PCI-bus is not available to the framegrabber (that means some other devices become master) are bypassed by a FIFO-memory. This allows only a short time span to bypass the blocked bus, since otherwise an overflow might occur and portions of the image are lost. Nevertheless, generally this is of no problem. The bus is controlled by the PCI-card's parameters *Maximum_Latency* and *Minimum_Grant* which are adjusted from the software driver automatically. If required, this parameters have to be adapted to the data transfer rate, to the system configuration and to the bus performance.

The *pciGrabber-4plus/express* is very flexible concerning the storage of the data. The user can specify destination and format of the data within a certain scope. For this a mechanism is required to separate the continuous flow of data into partitions and direct the data to the required addresses.

This mechanism is accomplished by the *pciGrabber-4plus/express* with the help of the *pixel instruction list*. This is a RISC-program, which drives the DMA-controller correspondingly.

This RISC-program has to be set up by the user application and must fulfill the required tasks and has to match the data and image format. So the program has to be specified according to each requirement, which implies that the RISC-program is created during run time of the

user application, since often the parameters (for example the size of the image) which control the RISC-program, are variable or not available prior to this time.

The software driver delivered with the *pciGrabber-4plus/express*, creates the appropriate RISC-program automatically with the adjustment of the image size. This process is transparent to the user application.

Nevertheless the application programmer should understand this principle of operation of the framegrabber, since this is reflected in the structure of the application software.

Figure 49 depicts this scheme. For image size and data format selection the user program applies the function *set_image()* of the driver. The driver starts two actions:

First the image size is set in the *VideoScaler* by values in the local registers of the framegrabber via the PCI-bus. This implies, that the *pciGrabber-4plus/express* creates the correct image size and the data flow has the correct format and provides the appropriate sync signals. In the same way the *DataFormatConverter* is adjusted to the correct format. This implies that the flow of pixel data to the FIFO has the correct format (for example RGB).

The second action of the driver software is the creation of a RISC-program appropriate to the data flow, which is stored in the main memory of the PC. The DMA-controller of the *pciGrabber-4plus/express* is notified of the start address of this program. During capturing, the DMA-controller fetches the RISC-commands in sequence by DMA from the main memory and processes and stores the data according to those instructions.

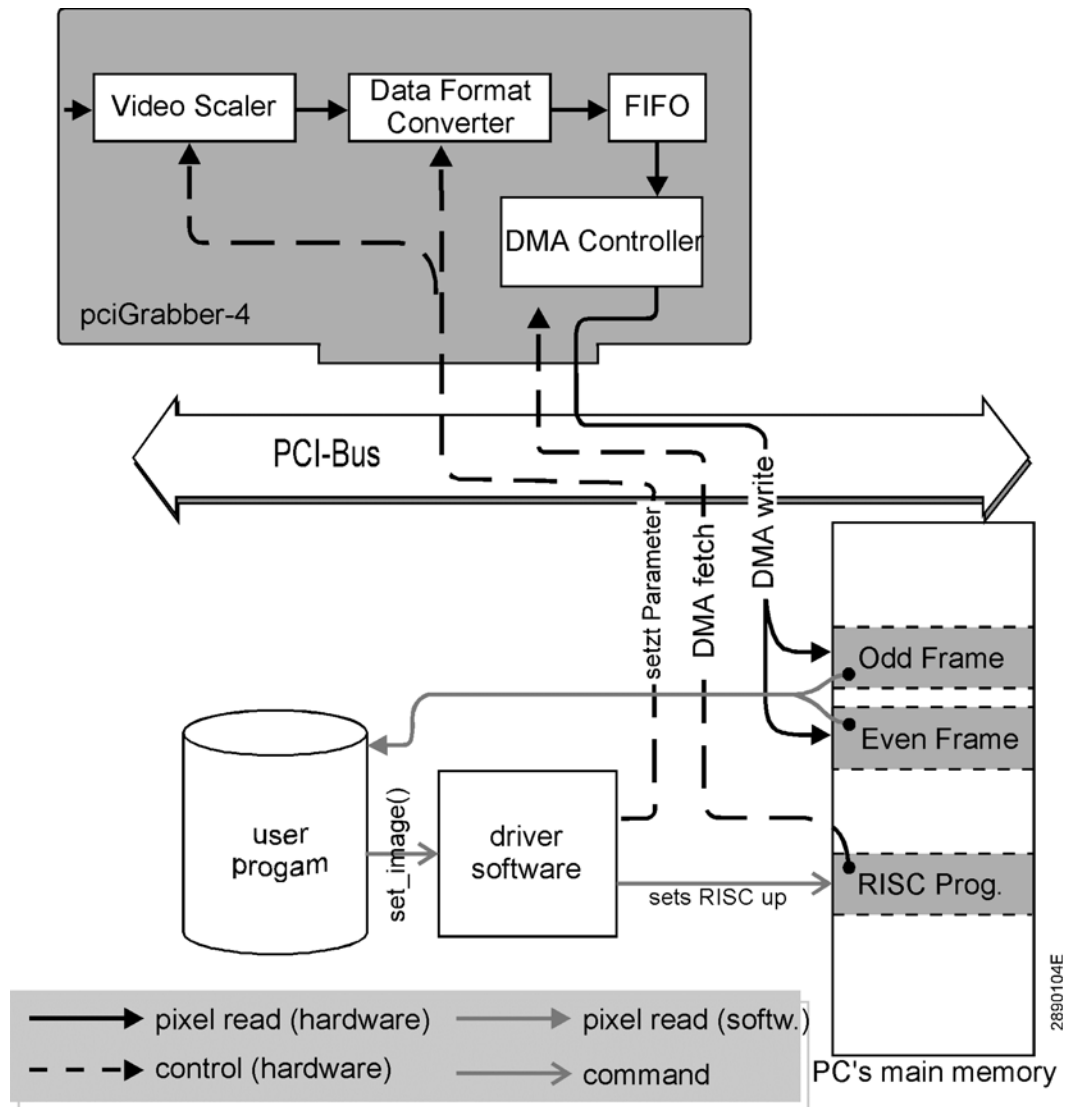


Figure 49: Pixel- and Control Data Flow (Overview)

By means of DMA-access the flow of data is directed to the main memory address specified by the RISC-program. This address region is reserved by the device driver (Windows). The application software can query the address range where the image data are stored (*GetPictureBufferAddress()*). With DOS, the software application itself can allocate the memory range for the image (e.g. by definition of arrays).

The regions might be defined - as shown in *Figure 49* - as two separate regions, one for the odd- and one for the even field, or only

one region, in which a whole frame is composed from the odd and the even field by the *pciGrabber-4plus/express*. The different options are selected by a parameter in the *set_image()* function, which influences the creation of the RISC program via the driver.

The driver provides information of the status, which indicates the end of the storage of the image in the memory, so that at this time all data are available.

This mechanism guarantees a fast access to the data. The process is real time regarding to the standard TV format. For the digitizing of a field it takes a time of 20 ms and the digitizing of a whole frame lasts 40ms. In addition, a time delay might have to be added to get the total time from the demand of the image to the end of the capture process.

This additional time might arise from waiting for the appropriate field. For example, if an even field is demanded, but the camera has just started scanning an even field, so it is necessary to wait until this field and the next following odd field are finished. In the worst case a delay of 40ms (two fields) can be expected. Now the demanded field can be digitized, which will last another 20 ms. During the following 20 ms nothing will happen in this memory region since the odd field will be received. The next 20 ms a new even field is stored in the memory if continues capturing is requested. It must be considered that the old information will be overwritten by the new information. This can lead to misinterpretation of the image content by the software especially for moving scenes.

Now we consider the case of capturing a whole frame in one memory region. Here the same effect might occur but in some different fashion. After 20 ms digitization the information for an odd field is completely available and therefore all odd lines are defined, but the even lines are not defined. During the following 20 ms the data for the even field are received. Immediately afterwards the new odd field will be digitized in continuous mode. Therefore there is hardly any time available to transfer data to the memory except for the blanking interval. So there is always a point (X,Y) where old and new information are stored adjacently, so that a mismatch will show up.

7.2 Driver for Microsoft Windows

The setup program for the Windows demo program copies all files needed on the computer's hard disk. The structure of the folders is similar to *Figure 50*. The window on the left-hand side shows the path to the folders. The path names can be edited during installation in order to create user specific names for the system. The libraries and *include* files needed to compile an own application are located in the corresponding subdirectories.

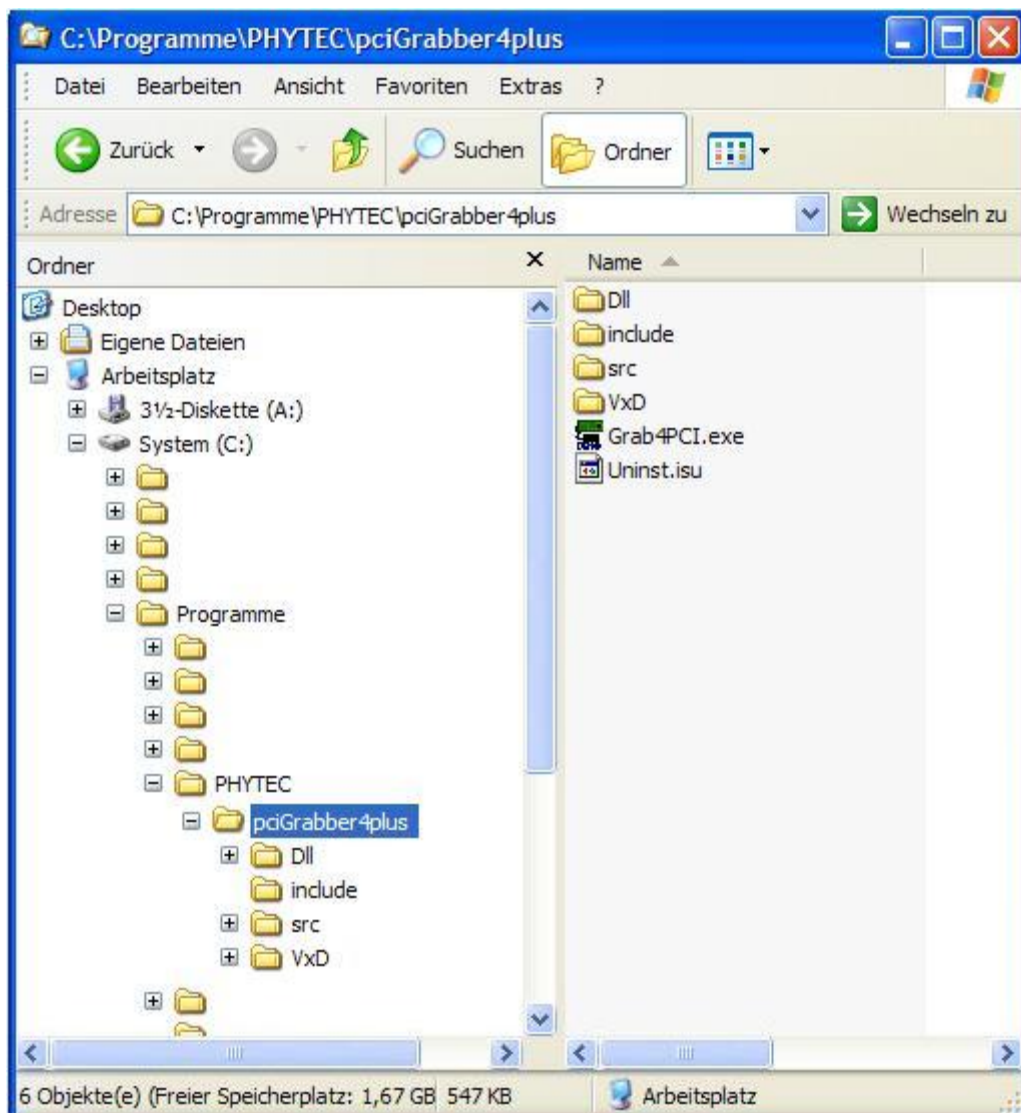


Figure 50: Folders for Window's Driver

7.2.1 Requirements

Applications for the *pciGrabber-4plus/express* can be created with the help of various development environments.

The users specific applications will work with the Windows95 / 98 / ME / XP / VISTA™ and Windows NT4.0 / 2000 operating systems.

Caution:

The device driver and corresponding DLLs must be copied into the Windows folder in order to implement the *pciGrabber-4plus/express* in a Window's operating system. In addition, the system driver must be registered into the registry table.

Phytec's setup software automatically copies the device driver and DLLs and registers the system driver. Therefore, all the requirements for operation are fulfilled.

Creating corresponding installation routines is recommended when installing your own applications onto other computers.

7.2.2 Installation of the VxD-Driver (Windows '95)

The VxD-driver has the purpose to allocate a continuous memory region, which will be the memory location for the image. Since the *pciGrabber-4plus/express* stores the data per DMA-access, it is required, that this physical fixed area has continuous addresses in a sequence.

The reservation of this memory region can only be achieved for Windows'95 by the means of a VxD.

By the VxD, the linear memory addresses are converted to physical memory addresses.

The VxD-driver is not called directly by the user program. The access will be executed via the DLL.

You might use the contents of the setup-CD to deliver the Windows 95 driver with your own application. The path on the CD is :

PCIGRAB4/DRIVER/WIN95_98.

You can copy these files onto your own setup media and distribute them with your application. Alternatively, you can create your own installation routine. In order to do this, the following steps must be followed:

The VxD-driver has to be registered in the Windows'95-system as *static device-driver* . This will be established in the following way:

- First the file `PCIGRAB4.VXD` must be copied to the Windows system folder.
- In the next step the driver will be included in the *registry* table : Please use the `REGEDIT` application in the `WINDOWS95` - folder. Please step through the registry key tree `HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services` until you reach `VxD` (see *Figure 51*).

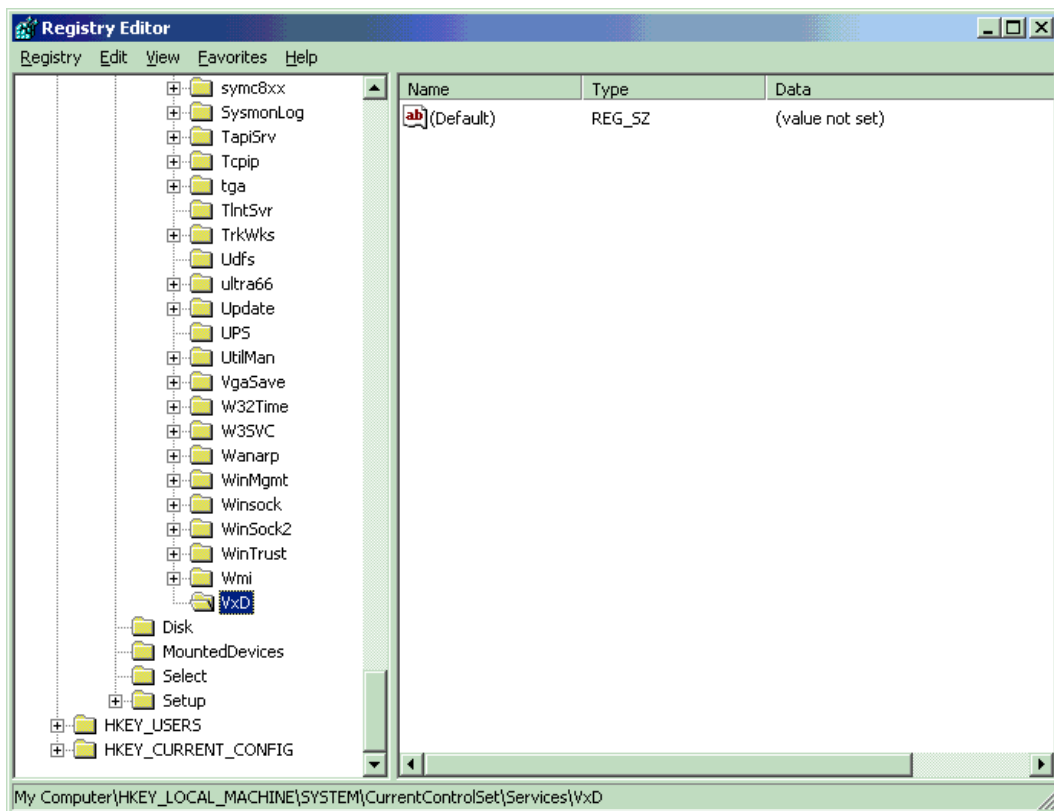


Figure 51: Windows'95 Registry Editor

Extend the key branch *VxD* with „Grab4PCI“, by selecting the menu “*Edit - New - Key*”, create a new key and assign the name “Grab4PCI” as shown in *Figure 52*.

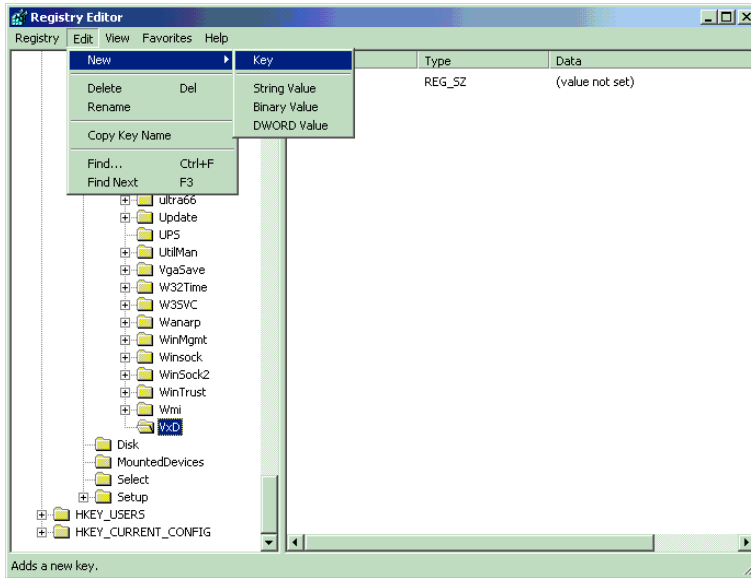


Figure 52: Adding of a VxD-Entry

Next please configure the new key branch : mark the listing „Grab4PCI“ as shown in *Figure 53*. Select from the menu *Edit - New*“the option „String Value“. Within the key of „Grab4PCI“ an entry will be generated with the notation „New Value #1“ . Please change this to „StaticVxD“. Click with the right mouse button the entry and select „Modify“. Write in the following dialog in the field „Value“ the character string „pciGrab4.VXD“ .

Select *Edit - Ne* with the option “Binary. Value“ and create as before the binary entry „Start“ with the value „00“. The final result must look exactly like shown in *Figure 53*.

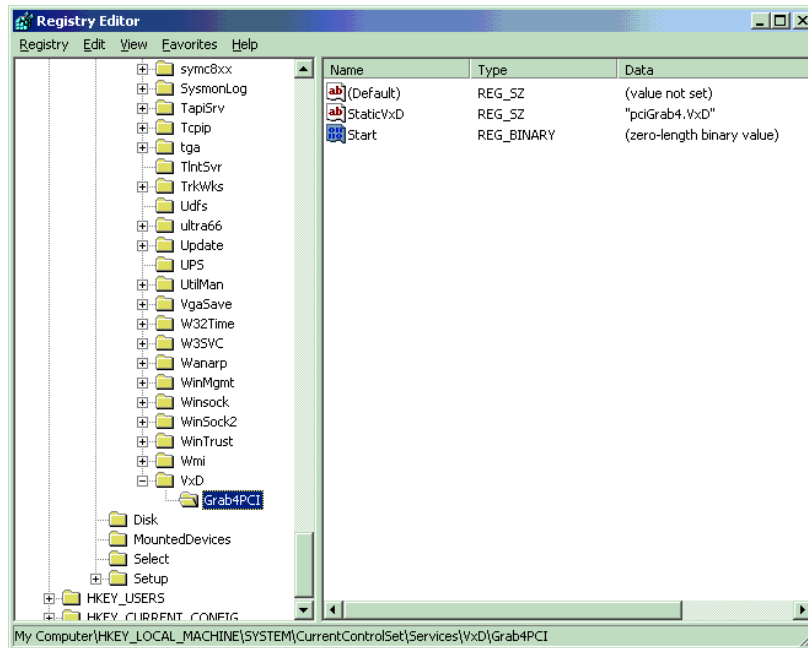


Figure 53: Setting Up the VxD

- Since Windows'95 is updating the changes of the registry only on booting, you'll have to start the system again.

Caution:

During deinstallation of the user application, the VxD-driver should be removed also. The driver entries have to be removed from the registry and the driver files should be deleted from the system folder. The VxD driver for the *pciGrabber-4plus/express* requires a region of 1.2 MB in the main memory, which is not available for other applications.

Please be very carefully to change the registry values, since a faulty alteration might destroy the whole configuration. Even your Windows'95 system might not be operating anymore!

Installation and deinstallation routines for the *pciGrabber-4plus/express* should be provided to the user, so that this process is to be executed automatically.

7.2.3 Application of the Device Driver for Windows NT4.0

The device driver functions in the same manner as with Windows 95; the driver allocates physical memory to store images. The driver also allows access to the framegrabber's register.

You might use the setup CD to deliver the Windows NT4.0 driver with your own application. The path on the CD is :

PCIGRAB4\DRIVER\WINNT40

You can copy these files onto your own setup media and distribute them with your application. Alternatively you can create your own installation routine.

When creating your setup routines, please carry out the following steps:

The driver has to be registered to the Windows NT4.0 system. This can be done in the following manner:

- The *PCIGRABBER4.SYS* file has to be copied into the folder *<Windows>\System32\drivers*.
- Next, the driver entry is added to the *Registry*:
Use the REGEDIT application (located in the WindowsNT directory). Scroll down the directory tree *HKEY_LOCAL_MACHINE/System/CurrentControlSet* and select *Services* (see *Figure 54*).

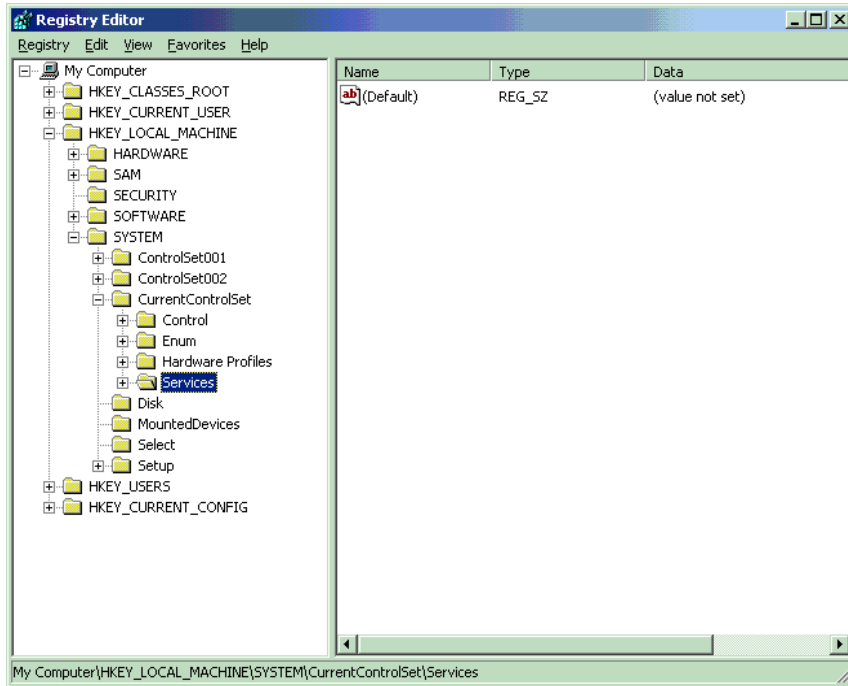


Figure 54: Windows NT Registration Editor

Open the *Services* folder. Select the *Edit/New/Key* pull-down menu and a new key will be created. Name this new key „*pciGrabber4*“, as shown in *Figure 55*.

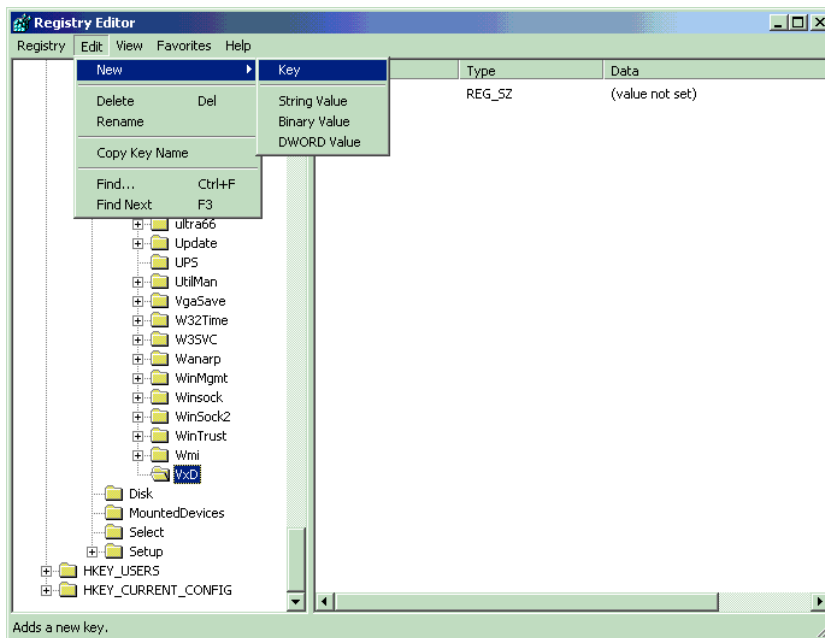


Figure 55: Entering a Device Driver

Now mark the entry with „*pciGrabber4*“ as shown in Figure 56 in order to configure the new key group.

Select the *DWORD value* command from the *Edit/New* pull-down menu. A new entry named „*New Value #1*“ will be created within the „*pciGrabber4*“ key.

Change this name in „*Start*“. Right click on the newly created entry and select *Modify*. In the dialog box that will appear, enter the number 2 into the *Value* field. Select the *DWORD value* command option from the *Edit/New* pull-down menu. Change the description in „*Type*“ and enter a value of 1.

Similar to the previous *DWORD* entry, select *DWORD value* from the *Edit/New* pull-down menu and enter a value of 1 for „*ErrorControl*“. The end result should look similar to *Figure 56*.

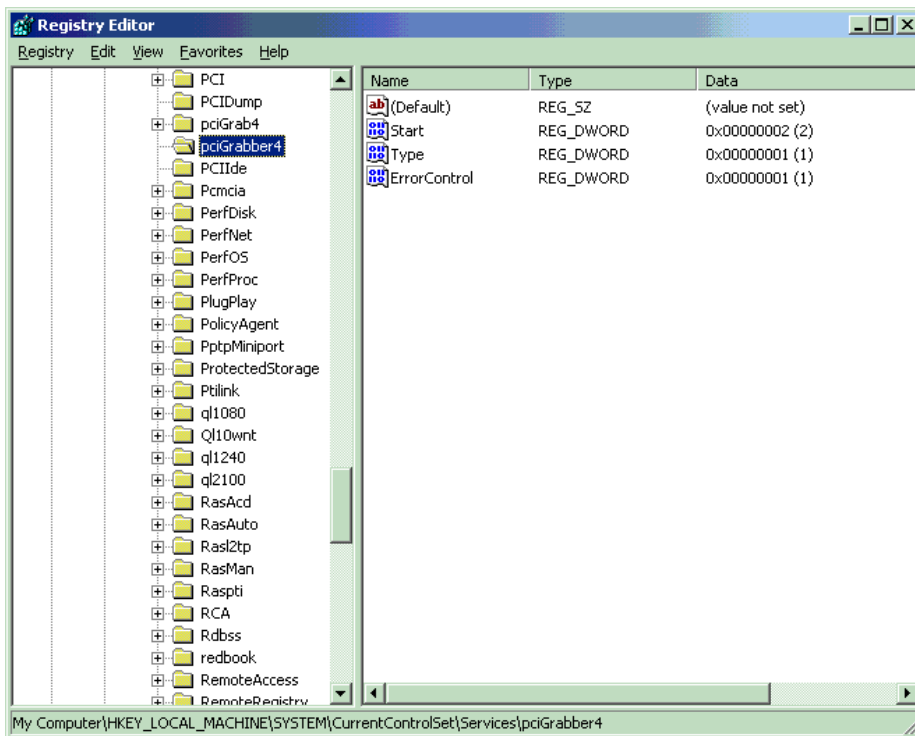


Figure 56: Configuring the Driver

After restart of the the computer, the driver is automatically loaded when starting Windows NT.

Caution:

During deinstallation of the user program the device-driver should be removed. The driver entry has to be removed from the registry and the files should be deleted from the system-directory.

The device-driver requires for the *pciGrabber-4plus/express* a region of 1.2 MB in the main memory, which is not available for other applications.

Caution:

Pay careful attention when changing the registry entries. If an error occurs while making these changes, the configurations can be permanently damaged. This could render the Windows NT operating system inoperable.

To the user, installation and deinstallation programs should be provided, so that this process is executed automatically.

7.2.4 Application of the Device Driver for Windows‘ 98™ and Windows‘ 2000 / XP / VISTA

Similar to Windows‘ 95, the device driver allocates physical memory for storing images. The driver also allows access to the framegrabber’s registers.

Reserving this type of memory space under the Windows98/2000 operating systems is only possible with a device driver.

The driver also transforms linear memory addresses into physical memory addresses.

User programs do not have direct communication with the driver, instead access is provided by DLLs.

You might use the installation CD to deliver the Windows98/2000/XP/VISTA driver with your own application. The path on the CD is ***PCIGRAB4\DRIVER\WIN2k_98***.

You can copy these files onto your own setup media and distribute them with your application.

7.2.5 Application of the DLL

The DLL provides communication between user applications and the *pciGrabber-4plus/express*. The DLL configures the framegrabber and controls capture events. In addition, the DLL allows access to the data of captured images stored in the main memory.

Caution:

The DLL is not linked into the user application, but called during runtime. Therefore, the DLL file must be available in the Windows system directory during runtime.

In addition to GR4CDLL.DLL the following DLLs are necessary for operation:

- **MSVCRT.DLL**
- **CTL3D32.DLL**
- **MFC42.DLL**

Windows provides various API functions to dynamically link the DLL. **Load Library(...)** is used to load the DLL and a handle is subsequently returned for the DLL. The API function **GetProcAddress(...)** provides starting addresses for various DLL functions. In order to release DLLs at program end, call the function **FreeLibrary(...)**. For more information, please refer to the development environment's User's Manual/Data Sheets or refer to the enclosed SDK source.

7.2.6 Application of the Windows 95/98/ME/XP/VISTA™ Windows NT4.0™ / Windows 2000™ DLLs

In order to use the DLL *Gr4CDLL.DLL*, the software developer must define a function pointer for each function that will be used in the application.

Example:

- Function to be used: **WORD Get_Error(void)**
- Definition of the function pointer:

```
WORD (PASCAL * lpfm_GetError)(void);
```

Use **GetProcAddress(...)** to obtain the relationship between the function pointer and the DLL.

Example:

```
lpfn_GetError = (WORD(PASCAL *)(void))  
                GetProcAddress(handle, „Get_Error“);
```

The function can now be called with:

```
WORD Errorstatus;  
...  
Errorstatus = lpfm_GetError();
```

Caution:

Check the value of the function pointer (return value from `GetProcAddress`) to be sure that it returns a 0 (zero). A value of 0 ensures that the driver version installed on the user's computer supports the functions and will return a valid handle.

7.2.7 Programming under Delphi

To introduce the functions to the DLL in Delphi, a corresponding *Unit* has to be defined. Please pay attention to the correct calling sequence, to guarantee the compatibility of the DLL. Define the functions with the type `stdcall`. In case of false declarations stack-overflows or underflows can occur, which will cause a violation against protected areas. In the following example a unit is defined:

```
unit grab4dll;

interface

{ The calling sequence 'stdcall' defines the sequence of the pa-
rameter transfer to the stack and signals to Delphi that the
called function frees the stack region, which was used for the pa-
rameter }

function Grab4_Get_Error: word;
    stdcall; external 'gr4cdll.dll' name 'Get_Error';

function Grab4_Max_Device_Number: word;
    stdcall; external 'gr4cdll.dll' name
    'Max_Device_Number';

function Grab4_Data_Present(nDevNo: word): word;
    stdcall; external 'gr4cdll.dll' name 'Data_Present';

function Grab4_GetPictureBufferAddress(nDevNo: word;
    dwBitsSize: Cardinal): cardinal; stdcall;
    external 'gr4cdll.dll' name 'Data_Present';

procedure Grab4_Initialize(nDevNo: word);
    stdcall; external 'gr4cdll.dll' name 'Initialize';

procedure Grab4_Set_Channel(nDevNo, nChannel: word);
    stdcall; external 'gr4cdll.dll' name 'Set_Channel';

procedure Grab4_Start_Grabber(nDevNo: word);
    stdcall; external 'gr4cdll.dll' name 'Start_Grabber';

procedure Grab4_Stop_Grabber(nDevNo: word);
    stdcall; external 'gr4cdll.dll' name 'Stop_Grabber';
```

```
procedure Grab4_Set_Image(nDevNo: word;
                          nOhpos, nOvpos, nOhsize, nOvsize,
                          nOppl, nOlines, nOColformat :word;
                          nEhpos, nEvpos, nEhsize, nEvsize,
                          nEppl, nElines, nEColformat :word;
                          nColsystem:word;
                          nInterlaced:word;
                          nSingleShot:word);
  stdcall; external 'gr4cdll.dll' name 'Set_Image';
```

```
const
  NTSC_M:    word    = 0;
  PAL_BDGHI: word    = 1;
  SECAM:     word    = 2;
  PAL_M:     word    = 3;
  PAL_N:     word    = 4;
  AUTO:      word    = 5;

  RGB32:     word    = 0;
  RGB24:     word    = 1;
  RGB16:     word    = 2;
  RGB15:     word    = 3;
  YUY2:      word    = 4;
  BtYUV:     word    = 5;
  Y8:        word    = 6;
  RGB8:      word    = 7;
```

implementation

{ DLL Functions }

end.

7.2.8 Description of the DLL's Functions

All actions of the *pciGrabber-4plus/express* are initiated using the functions of the DLL. Also, the DLL is used to read the actual status and to configure the framegrabber. These functions are described in more detail further on in this manual.

The functions have been divided into five groups for easier discussion. The groups are numbered, for reference the group number is shown in a black circle on every function.

The functions are classified as follows:

❶ Routines for initialization / check of hardware status

This group includes routines that must be called once prior to using the framegrabber, to ensure that the framegrabber functions properly. Also included are two functions that give information about the hardware installed and its capabilities.

❷ Routines that configure the framegrabber for the capturing

Functions from this group configure the framegrabber to the connected image source (camera). These functions also determine the appearance of the captured picture in memory (image size, color format, etc.) The user should determine whether each function is needed for his purposes, and which parameters are necessary. These functions may be called several times (i.e. when the input channel should be switched or if the image size should be changed).

❸ Routines for initiating and controlling the image capture

These functions start the image capturing, monitor the capture process and end the process.

❹ Routines for configuring image parameters


Functions from this group enable the configuration of parameters, such as brightness, contrast, saturation, etc. These functions are not necessary for capturing, but can be called at any time to adapt the appearance of the final image to user needs.

⑤ **Routines for controlling the option port and other features**

This category includes functions that do not directly influence the capture process, but rather deal with the features of the framegrabber, i.e. I/O port, I²C interface, etc. These functions need only to be called when a corresponding framegrabber feature is to be implemented into the application software.

Most functions are identical in the Windows and in the DOS driver versions. Although, depending on driver type, differences will occur. In order to simplify the porting process, the following symbols are used:

↔ Calling functions are the same under DOS and Windows.
Nevertheless, please note that the variable types can differ depending on the operating system.

 This function is specific to Windows

DOS This function is specific to DOS

Caution:

In all of the following descriptions for routines, the parameter `nDevNo` is used. This parameter identifies the desired *pciGrabber-4plus/express* when multiple Grabbers are installed in the system. The number of installed Grabbers can be determined by the function **Max_Device_Number()**.

Compatibility to the pciGrabber-4

The driver is downwards compatible with the pciGrabber-4 (VD-007 and VD-007-X1). Programs that have been written for the pciGrabber-4 also function with the pciGrabber-4*plus/express*.

In order to ensure operation of new features for the pciGrabber-4*plus/express*, new or altered functions have been created for the driver.

Functions that are not compatible with the older driver version for the pciGrabber-4 are denoted with a star (☆).

Please take note of the functions with a ☆ when adding new features from the pciGrabber-4*plus/express* to existing applications.

Most functions are compatible with the pciGrabber-4, although some functions may not be used due to non-compliance with hardware requirements. In any case, the new driver version should be used with new applications.

⇔ ❶ Evaluation of the Error Messages

WORD `Get_Error(void)`;

Return value:

- 0 = no error
- 1 = device number not found
- 2 = bad register number
- 3 = initialization failed
- 4 = Grabber not found
- 5 = unknown parameter value
- 6 = not supported
- 7 = newer driver version required (update)
- 8 = no PHYTEC grabber card found
- 9 = no acknowledge
- 10 = invalid address
- 11 = write access denied

Each call of a driver function should be checked if it was successful using the function *Get_Error*. Immediately after the call of the function, the internal error variable of the driver is set to the actual status.

This variable is available to the user program via the function *Get_Error*, so that the application can react in a corresponding way to this error message.

The investigation of the error variable is possible until a new call of a driver function has occurred. Then the error status will reflect the success of the new function call.

❶ Obtaining the Version Number of the DLL

DWORD GetVersionNumber (void);

Return value: Version number for the Grab4CDLL
 HighWord: Major_Version_Number
 LowWord: Minor_Version_Number

The version number for the Grabber-DLL can be obtained using this return values.

This function is only available with the Windows-DLL and not for DOS applications.

Note:

Test the version number and ensure that when using the *pciGrabber-4plus/express* the Major_Version_Number is larger than or equal to 4.

⇔ **① Determination of the number of available
pciGrabber-4plus/express**

WORD Max_Device_Number (void);

Return value: Number of pciGrabber found

This functions specifies the number of *pciGrabber-4plus/express* in the system. This is required, because PCI-/PCI Express-devices are not configured with jumpers or other hardware, but are assigned automatically an address region, by the PCI-BIOS. With the help of the PCI-BIOS, the address region can be determined for each framegrabber card. If several cards are installed in the system, the addresses are returned in a sequence (*see also 3.4 and 4.4*).

The user has not to care about addresses or address regions when using the driver. Those are converted internally into *device numbers* (= *nDevNo*). Each *pciGrabber-4plus/express* card in the system is assigned a unique device number. It can not be predicted which number is assigned to which card, since this depends on the topology of the bus and the function of the BIOS.

Note:

If an identification of each physical card is needed, the DIP-switch option of the *-RS* models might be helpful.

In order to access the different *pciGrabber-4plus/express* independently by the software, the device number is transferred as parameter with each function call.

The function `Max_Device_Number()` is applied to find out how many *pciGrabber-4plus/express* are installed in the computer. The highest permissible device number is returned. At the same time this is the number of grabbers in the system, since the lowest device number =1. If the returned value is 0, the PCI-BIOS did not find a *pciGrabber-4plus/express* installed in the computer.

For *nDevNo* only values between

$$0 < nDevNo \leq \text{Max_Device_Number}()$$

are accepted.

↔ ❶ Initialization of the Grabber and driver after activation

void Initialize(WORD nDevNo);

This routine initializes the framegrabber and the driver software after turning on the system. This routine **must** be called before the first access to the framegrabber. This initialization must be carried out for each separate framegrabber to be used. This means, that *Initialize* has to be called for all permissible values of `nDevNo`.

↔ ❷ Obtaining Information about the cards installed

short Read_GrabberInfo(WORD nDevNo, WORD wInfoType)

`wInfoType`: specifies which Grabber feature will be called

return value: Value of the specified feature

This functions delivers information on the hardware, in order to ensure optimal adaptation of the applications to the framegrabber.

This function can also be used to find out the number of input channels that the framegrabber will support so the channel selection dialog can carry out a range test.

The properties are returned with numerical values (WORD type). The meanings of the key numbers are described in the entries of the header file.

`wInfoType` can be used to select which information will be called. A description of the parameters is also included in the Header file. If a parameter is not defined, then the function returns a value of -1. The error status will return the value 6 = „NOT_SUPPORTED“.

The following parameters can be called:

GRABBER_TYPE: Specifies the type number of the framegrabber. The return value is numeric, and the Header file includes a description. For example if `Read_GrabberInfo (1, GRABBERTYPE)` returns 1, framegrabber #1 is a VD-009 model.

MAX_CHANNEL : Returns the number of available composite input channels. For example, for VD-009 = 9, for VD-009-X1 = 3.

Note:

The features that can be queried might be expanded with newer card models. Any available information can be found in the header file.

⇔ **1 Get Grabber Name as a Text String**

WORD `Read_OrderCode (WORD nDevNo,`
unsigned char* `sCodeString,`
DWORD `dwSizeOfString)`

***sCodeString:** Pointer points to a declared string (25 Bytes min). The function writes the framegrabber name into this string.

dwSizeOfString: Size of the reserved array

return value: Error Code

This function allows the framegrabber's description to be read in clear text. A null-terminated string stores the name. In order for the name to be transmitted, a character array must be reserved and a pointer must be handed over to the array in `*sCodeString`. The available size of the array is given by the parameter `SizeOfString`.

The size should be at least 25 characters.

If the type description does not fit in the reserved array, the function returns error code 5.

The type description corresponds to the order number. If a card does not have any clear text information available for the driver, the string „TYPE CODE=xx“ is returned. Xx = encoded type number (see *Read_GrabberInfo*).

Note:

If the pciGrabber-4 (VD-007) or a related product is called up, then the error code 6 and the string „VD-007 or compatible“ is returned.

↔ ② Grabber setting to the desired color-system

void Set_Color_System(WORD nDevNo, WORD nColSys);

nColSys: Code for color system

With the function **Set_Color_System** the framegrabber is configured for the color system used. Clock frequency and input registers of the video processor are set accordingly. The user can select for *nColSys* predefined constants:

- PAL_BDGHI configures the framegrabber for the application of PAL-video sources.
- NTSC_M configures the framegrabber for NTSC-sources

⇔ ② Reading video format

WORD Get_Video_Status(WORD nDevNo);

return value: 0 = 525 line format (NTSC / PAL-M)
1 = 625 line format (PAL / SECAM)

This function provides the recognition of the video format of the camera at the selected channel, and distinguishes if the source is a NTSC- or PAL/SECAM-system. The distinguishing feature is the different number of lines for both TV-systems. For the recognition, it takes 32 consecutive fields from applying the image source, until the identification is finished.

⇔ ② Configuring the Composite Mode (Composite Inputs)

void Set_Composite(WORD nDevNo);

Calling this routine switches the framegrabber into the composite mode. The chroma ADC is switched off and the luma notch filter is activated. This mode must be configured when composite signals are to be digitized.

The composite mode must be selected for all standard cameras (monochrome or color) which do not have an S-Video output to connect to the framegrabber. Composite cameras are connected to the framegrabber via i.e. the WK-012 and the WK-022 cables or the BNC connectors. The composite mode is configured during standard initialization.

Note:

After calling *Set_Composite*, the corresponding input channel must be set by calling *Set_Channel*.

↔ ② ☆ **Configuring the S-Video Mode****BYTE Set_S_VideoEx(WORD nDevNo, BYTE input);**

input: MINIDIN = selects Mini DIN socket for input
 COMBI = selects Combi-Input (HD-DB-15 connector)
 AUTO = automatic configuration

Return value:

(a) *input* = MINIDIN or COMBI

SUCCESSFUL = no error occurred

NOT_SUPPORTED = pciGrabber-4 (old model) and
 the COMBI parameter

(b) *input* = AUTO

MINIDIN = signal input set to Mini-DIN socket

COMBI = signal input set to COMBI socket

NO_SIGNAL = no signal found (AUTO mode)

The video processor's second ADC must be activated when connecting an S-Video source. The function **Set_S_Video** activates this *chroma* ADC. Also, the luma notch filter in the Luma path is deactivated, since it is not needed for S-Video signals. This results in a sharper image. **Set_S_Video** also automatically connects the input channel to the S-Video input.

S-Video sources are color cameras that have a special output in order to separate brightness and color signals. These cameras can be connected to a framegrabber with either the WK051 or the WK075 cable.

Caution:

An S-Video source can either be connected to the Mini DIN socket or to the (lower) HD-DB-15 socket.

Multiple S-Video sources may not be connected to both sockets simultaneously!

The function has a parameter that specifies which socket the S-Video camera is connected to. If AUTO is given as a parameter, the driver searches for the S-Video signal automatically. The driver at first checks the Mini DIN socket for an active signal.

If a signal is found, the framegrabber is configured to the Mini-DIN socket and the parameter `MINIDIN` is returned. If an active signal has not been found, then the framegrabber tests the S-Video connector on the Combi socket (HD-DB15-socket ②). If an active signal is found on the Combi socket, then the framegrabber is configured to the combi socket and the parameter `COMBI` is returned.

If a signal has not been found at either of the sockets, then the framegrabber is configured to the Mini DIN socket and the return value is `NO_SIGNAL`.

Note:

- If there is no S-Video source connected, but there is a composite source present at channel 9 (VD-009), or channel 3 (VD-009-X1/VD-011), the framegrabber is configured to the Combi socket by the AUTO function. In this case, the image of the composite source is displayed in black and white (monochromatic).
- The AUTO function does not work if the connected video source does not supply a video signal.
- This function is not compatible with older driver versions.

↔ ② ☆ Configuring the Input Channels

void Set_ChannelEx(WORD nDevNo, WORD nChannel);

nChannel: Input channel to be configured (1...9 / 1...3)

This function is used to select the input channel for composite sources. The signal is switched by a two stage multiplexer. The first stage is externally located on the framegrabber board, and the second stage is integrated in the video decoder chip.

Values:

VD-009: Channel numbers 1-9 allowed

VD-009-X1: Channel numbers 1-3 allowed

VD-011: Channel numbers 1-3 allowed

This function is not compatible with the pciGrabber-4 (VD-007).

Note:

Configuring the input channels is not necessary when using S-Video sources! The function *Set_S_VideoEx()* switches the channels automatically.

Caution:

When switching input channels, latency times have to be considered until an image for the new channel can be captured. This has several reasons explained below.

- Because of the definition of the video signal it normally takes up to 4 fields before synchronization is established and the color decoding functions correctly.
- Due to DC voltage decoupling between the framegrabber and the camera, different average DC levels on the signal lines are present. This can cause charge transfer effects while switching over.
- The framegrabber's AGC must first lock to the new signal level.

- It is not possible to change channels from one image to the next. The user should adhere to latency times of at least 80 ms. This is an approximated value which is also dependant upon the signal sources connected.

Note:

If the S-Video input is not being used, it is possible to use an additional composite channel. This allows a total of 10 composite inputs for VD-009 and 4 composite inputs for model VD-009-X1 / VD-011. In order to activate additional channels, channel number 10 (VD-009) or 4 (VD-009-X1/VD-011) is handed over to the function *SetChannel*. S-Video sources may not be connected to the framegrabber and it should not be switched to S-Video (this means that the function *Set_S_VideoEx* should not be called).

The additional composite input is available at the following sockets:

VD-009: first HD-DB-15 socket, pin 4
VD-009-X1: additionally at the lower HD-DB-15 socket, pin 3
VD-011: additionally at the HD-DB-15 socket, pin 3

Pins 5- 8 can be used as signal Ground.

The function also supports the older models VD-007 and VD-007-X1. When using the older models, the following parameters are allowed:

VD-007: Channel numbers 1- 9 allowed
VD-007-X1: Channel numbers allowed:

1 = input 1
5 = input 2
9 = input 3

↔ 2 Selection of the luma notch filter for black/white operation**void Set_BW(WORD nDevNo, WORD nOn);**

nOn: 0 = composite-signal at the input
(activate luma notch filter),
1 = b/w-signal at the input (deactivate luma notch filter)

If a black/white (monochrome) camera is connected to the frame-grabber, a luma notch filter (which avoids disturbing color moiré from the brightness signal) is not necessary (*Cross-Color-Effect*).

This function allows the activation and deactivation of the luma notch filter by software. For b/w operation the deactivation of the luma notch filter is advisable, because the sharpness of the image can be improved this way. In standard mode, the luma notch filter is activated.

⇔ ② De-/Activation of the Interlaced Mode

void Set_Interlace(WORD nDevNo, WORD nInterlace);

nInterlace: 0 = Non-Interlace
 1 = Interlace
 2 = Field Aligned

This function indicates to the framegrabber, that the incoming video signal is an interlaced or none interlaced signal. This will influence the vertical scaling filter. For frame capturing the option *Interlace* and for a field the option *Non-Interlace* should be selected in order to reduce artefacts from the motion.

When displaying only fields, a 20 ms pause usually occurs between two capture events. This is because only one field of the two fields that form a frame can be used due to interlacing system. In the *Field Aligned* mode, the second field is internally shifted by a half row, so that it fits onto the first field. This method allows capturing a field every 20 ms.

Because the field is processed electronically by the framegrabber, this function might not return images suitable for measuring and automation tasks.

↔ 2 De/Activation of the AGC

```
void Set_AGC(WORD nDevNo,WORD nCAGC, WORD nAGC,  
             WORD nCrush);
```

nCAGC: 0 = chroma AGC off
 1 = chroma AGC on
nAGC: 0 = AGC on
 1 = AGC off
nCrush: 0 = none-adaptive AGC
 1 = adaptive AGC

The *pciGrabber-4plus/express* contains two AGCs (Automatic Gain Control). The common AGC controls the signal level of the composite signal (Y), correspondingly the input amplitude.

In addition, the Chroma AGC controls the adaptation of the amplitude of the color signal.

For the AGC the modus *Adaptive AGC* is available. In this case the overflow bit of the A/D converter is monitored. If an overflow occurs, the A/D reference voltage is automatically increased, which causes an increase of the input voltage range.

In general, operating the *pciGrabber-4plus/express* with a non-adaptive gain control will be sufficient.

In some applications the adaptive AGC might cause disturbances (e.g. when working with absolute brightness values is mandatory). In this case, non-adaptive AGC should be used.

When using applications that switch between multiple cameras (i.e. video surveillance), the time needed to switch between cameras can be reduced, under certain circumstances, by using adaptive AGC.

⇔ **2 De/Activation of the color killer**

void Set_CKill(WORD nDevNo, WORD nCKill);

nCKill: 0 = color killer off
1 = color killer on

In case of b/w signal sources are connected to a color system, color noise might be visible. The color killer eliminates this effect, by testing if the color burst signal is present and in case it is not, deactivating the color stage.

It might be desirable to capture a color signal with a weak color carrier. Due to the signal quality, the color killer might switch to monochrome mode and color recognition is not possible.

With *nCKill* = 0 the color killer can be turned off, so that the image is captured with color information, but some color noise might appear with weak signals.

By default, the color killer is on, and switching between color- and monochrome sources is done automatically.

↔ 2 Dropping fields/frames in a video Signal

```
void TemporalDect(WORD nDevNo,  
                  WORD nDecField,  
                  WORD nFldAlign,  
                  WORD nDecRat);
```

nDecField: 0 = drop frame(s)
 1 = drop field(s)

nAlign: 0 = odd field will be dropped first
 1 = even field will be dropped first

nDecRat: number of the fields / frames to be dropped out of 50 (PAL)
 or 60 (NTSC)

The *pciGrabber-4plus/express* allows for continuous capturing to select the number of images digitized per second. Default is 50 (PAL) or 60 (NTSC) images per second (video-standard).

With the help of this function it can be set, how many images of the 50 or 60 images are dropped during digitization to decrease the result. The other two parameters give instructions of the kind of omissions: The parameter *nAlign* aligns the start of the decimation with an even or an odd field.

The parameter *TDecField* defines whether decimation is by fields or frames.

Example (PAL/SECAM):

- *nDecField=0, nDecRate=2*

The reduction refers to frames. From 50 images two are omitted. The images 1-24 are captured as usually, then an image is omitted. Images 26-49 are captured before again one image is omitted.

- *nDecField=1, nDecRate=25*

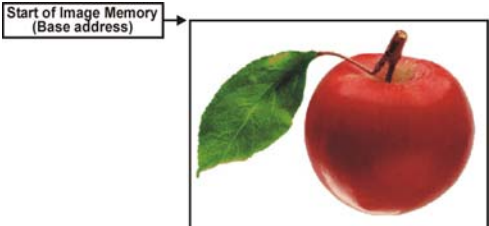
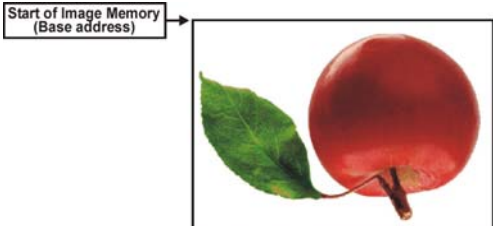
In this case 25 fields are omitted of 50. The result will be, that each second image will be captured, so always the same field type will be omitted. Which field will be omitted first, depends on the *nAlign* setting.

🏠 ☆ ② Flip Image Vertically

DWORD FlipPicture(WORD nDevNo, unsigned char flip)

flip : 0 = image is stored upright
 : 1 = image is flipped vertically (default)

With this function, the vertical orientation of the image in the memory can be set:

flip = 0	flip=1 (default)
<p>The image is stored upright. This means, the lowest image memory address contains the <u>upper</u>-left corner of the image:</p> 	<p>The image is flipped vertically before it is stored in the image memory. The lowest image memory address contains the <u>lower</u>-left corner of the image:</p>  <p>This is useful, if image data are processed in BMP-format.</p>

Important:

- The setting of the vertical image orientation with *FlipPicture()* must be done before the *Set_Image()* function is called. The settings take effect not before *Set_Image()* is called. To change the settings while the grabber is running, it has to be stopped first. Then the setting can be changed by calling the command sequence *FlipPicture()* and *SetImage()*.
- The default setting is **flip=1** (the image is stored upside-down; for compatibility reasons.)

② Setting the size and scaling of the image

```
void Set_Image (WORD nDevNo,
               WORD nOhpos, WORD nOvpos,
               WORD nOhsize, WORD nOvsize,
               WORD nOppl, WORD nOlines,
               WORD nOColformat,
               WORD nEhpos, WORD nEvpos,
               WORD nEhsize, WORD nEvsize,
               WORD nEppl, WORD nElines,
               WORD nEColformat,
               WORD nColSystem,
               WORD nInterlaced,
               WORD nSingleShot);
```

nOhpos, nOvpos : position of the upper left corner of the odd field within the captured field (used for cropping) (hpos = horizontal, vpos = vertical)

nOhsize : size of the odd field (X-direction)

nOvsize : size of the odd field (Y-direction)

nOppl : horizontal resolution of the odd field (ppl = pixel per line)

nOlines : vertical resolution of the odd field (number of lines)

nOColformat: color format: (RGB32, RGB24, RGB16, RGB15, Y8, YCrCb 4:2:2, YCrCb 4:1:1)

nEhpos, nEvpos : position of the upper left corner of the even field within the captured field (used for cropping) (hpos = horizontal, vpos = vertical)

nEhsize : size of the even field (X-direction)

nEvsize : size of the even field (Y-direction)

nEppl : horizontal resolution of the even field (pixel per line)

nElines : vertical resolution of the even field (number of lines)

nEColformat: color format: (RGB32, RGB24, RGB16, RGB15, Y8, YCrCb 4:2:2, YCrCb 4:1:1)

nColSystem : code for color system (see *Set_Color_System*)

nInterlaced : 0 = Non-Interlace
1 = Interlace
2 = Field Aligned

nSingleShot : 0 = continuous capturing
1 = **one** single image is captured

The routine *Set_Image* () defines the size, the position and scaling of the image sections delivered by the framegrabber separately for odd and even fields. In addition, the data format in which the picture will be stored later in the memory, will be defined.

Please note that this function has different parameters in the DOS version of the driver (see below).

Caution:

This function may be called only while the framegrabber is in Stop mode. *Set_Image* may not be called when the framegrabber is capturing, or when a Single-Shot digitization is not terminated by a call of the *Stop_Grabber* function.

The settings for both fields can be established separately and the parameters have a letter prefix 'E' = even field or an 'O' = odd field. Parameter without those letters are valid for both fields.

For both fields different sizes can be stated. They are be stored in different memory regions and can be processed in different ways. For example, one field can be used for showing a small preview image on the screen while the other field is used to process the data in higher resolution.

In *Interlaced-Mode*, both fields are sequentially interlaced and stored in one common memory region. This mode provides the maximum resolution of 720 x 576 pixels (PAL) or 640 x 480 pixels (NTSC), respectively. (See also the *section De-/Activating the Interlace Mode*)

In the following we consider the setting of the parameters without the field specifications (for example *hsize* for *nEhsize* / *nOhsize*). Please correspondingly precede the letter 'E' for 'Even' or 'O' for 'Odd'. If a specific field is required, it will be indicated as such.

In the first step a window with the *size* of the image is defined. This is achieved by the parameter *hsize* and *vsize*. *Hsize* indicates the number of pixels of the captured image in x-direction, and *vsize* the number of pixels in y-direction.

Next, the *resolution* of the image is defined. The value *ppl* and *lines* define how many pixels are generated from the incoming video signal. *ppl* indicates the number of pixels per line, and *lines* determines how many lines (pixels in y-direction) are produced.

An entire frame in PAL format has 720 pixel x 576 lines. In order to digitize the image with the highest resolution, *ppl* will be 720 and *lines* = 576. The smallest resolution allowed for PAL is 50 x 40 pixels per frame.

If fields are used, the maximum resolution is 720 x 288 lines.

Note:

It's important to understand, that the definition of *ppl* and *lines* **always refers to fields**. The actual number of lines of the video picture to be digitized is twice as high. The width/height ratio for the digitized *field* (maximum 720 x 288 pixels) gives a distortion of two in y-direction. In order to avoid this effect, digitization can be performed in interlaced mode (which will double vertical resolution) or the horizontal resolution can be reduced to *ppl*=360, *lines*=288, which will result in a proportional image of 360 x 288 pixels (see examples below).

Note:

For automatization applications, a correct proportional resolution is not always required, as long as the distortion is taken into consideration in the algorithm. So the complete field resolution of 720 x 288 can be used for stereometric work, which is more accurate in X-direction than in Y-direction. It might also be possible to adjust the axis of the camera by 90° in the direction of the measurement line.

The window with the proportions *hsize* x *vsize* is a section of the digitized image, which has the size *ppl* x *lines*. If *hsize* = *ppl* and *vsize* = *lines* the whole digitized image is displayed. If the parameters *hsize* x *vsize* are smaller, only a section of the image is displayed. The ratio of *hsize* to *vsize* does not alter the proportions of the image, since no scaling occurs but a certain section is cropped out of the image. *Figure 57* and *Figure 58* demonstrate how the parameters work.

vsize and *lines* always have to be set in respect to a field in case a field was digitized.

If the window defined by *hsize* and *vsize* is smaller than the size of the area determined by *ppl* and *lines*, the window can be moved in the digitized image by the parameters *hpos* and *vpos*. For *hpos*=0 and *vpos*=0 the window is positioned in the upper left corner of the captured image.

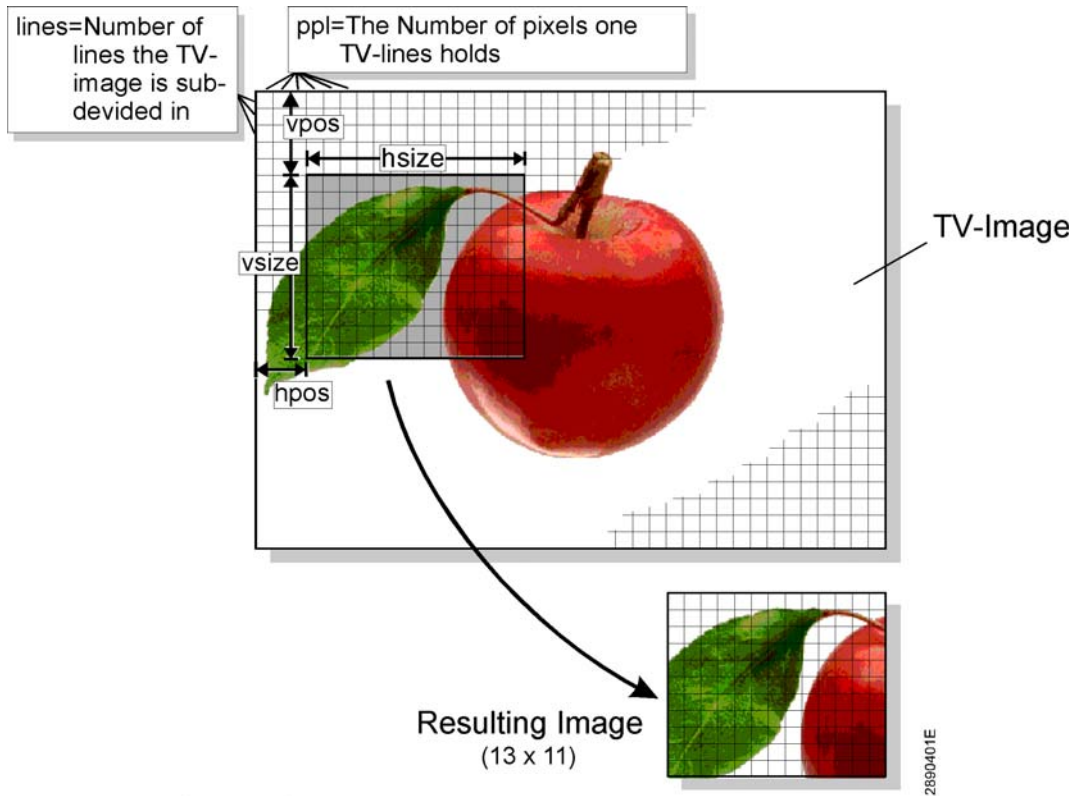


Figure 57: Scaling and Cropping

Caution:

- The area of the window defined by *hsize* and *vsize* and with the position *hpos* and *vpos* may not leave the region of the digitized image defined by *ppl* and *lines* :

hpos=100, *hsize*=200, *ppl*=200 : not allowed since the last 100 pixels are not defined

hpos=0, *hsize*=200, *ppl*=202 : allowed: all pixels are inside the captured image area

hpos=100, *hsize*=100, *ppl*=200 : allowed

hpos=100, *hsize*=200, *ppl*=300 : allowed

hpos=300, *hsize*=300, *ppl*=800 : not a allowed: image has more pixels than the TV-standard provides

(correspondingly in Y-direction)

- All parameters in horizontal direction must have **even** values. (*ppl*=123 is not permitted, *ppl*=124 is allowed.)
- If all parameters, which define the size of the image, are set to 0, no field is produced.

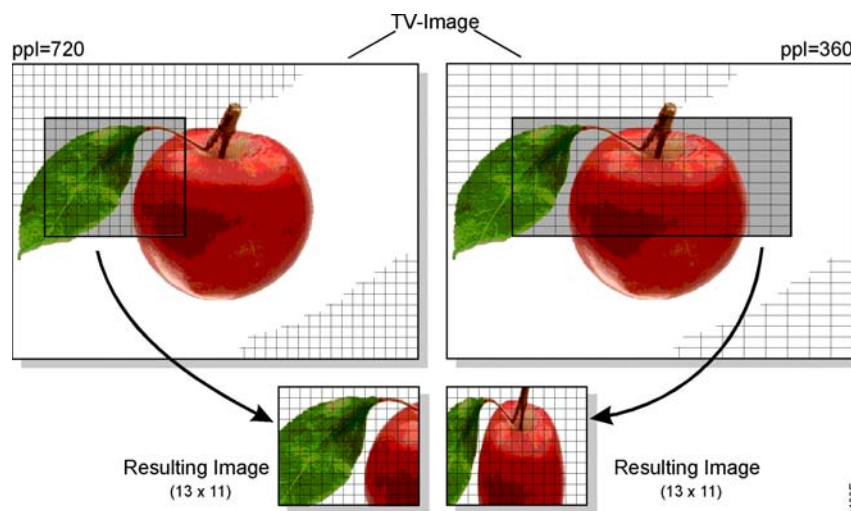


Figure 58: Example of Scaling: Only the *ppl* value is changed

Examples:

● ***field-based capturing***

A square image of the size 256 x 256 pixels is to be digitized with a resolution and proportion corresponding to the TV picture.

(a) Resolution and Scaling

For the desired capture window a field resolution (288 lines) is sufficient. In order to obtain a correct width/height ratio, the resolution in X-direction has to be reduced to the half (since a field = half height).

Thus, horizontal resolution is $720:2 = 360$.

The result is : $ppl=360$, $lines=288$

(b) Size of the section

The image should have a size of 256 x 256 pixels.

This results in $hsize=256$, $vsize=256$

(c) Positioning

It is advisable to center the section of the image.

In x-direction only 256 pixels of 360 pixels are displayed in the window. Outside the window $360 - 256 = 104$ pixels remain, which are divided in equal parts to both sides, which result in 52 pixels on both sides. $hpos$ is the size of the left edge, so $hpos=52$.

Correspondingly in Y-direction: $(288-256):2=16$; $vpos=16$.

Note:

It would be wrong to set $ppl=256$ and $lines = 256$. In this case the width/height ratio (TV-Standard 4:3) would be changed to 1: 1 and the image would be distorted. It would be possible to set $lines = 256$ and to compute ppl by the width/height ratio. In this case we would gain the optimal height of the image.

- ***field-based capturing with zoom***

An image of the size 120 x 100 is to be captured, for which the original image is zoomed out with a zoom factor of 2 in X- and Y-direction.

(a) Resolution / Scaling

It is sufficient to digitize a field. Without zoom 360 x 288 pixels would be used. In order to achieve the zoom factor, we set the resolution to the half: 180 x 144 pixels : $ppl=180$, $lines=144$. Note that the width/height ratio is maintained.

(b) Size of the Section

Corresponding to the size of the window we set:
 $hsize = 120$, $vsize = 100$.

(c) Positioning

With the parameter $hpos$ and $vpos$ the window section can be shifted $180 - 120 = 60$ pixels in X-direction and $144 - 100 = 44$ pixels in Y-direction.

- ***Full Frame Capture***

A 700 x 500 image should be captured with resolution and proportions that correspond to a TV image (PAL).

(a) Resolution and Scaling

For this example, a frame is required. Since a full TV image serves as the basis the image resolution results in $ppl=720$ and $lines=576$. The $lines$ value in the vertical direction must be evenly distributed to both half frames.

$$nOlines = nElines = \frac{1}{2} \text{ lines}$$

$$nOlines = \frac{1}{2} 576 = 288$$

$$nElines = \frac{1}{2} 576 = 288$$

Because the entire image has a total of 576 rows, 720 pixels are required for the X-direction, so that the height-to-width ratio is maintained.

$$nOppl = nEppl = 720$$

(b) Window Size

The image should be 700 x 500 pixels, resulting directly in an $hsize = 700$.

To maintain the height-to-width ratio, the pixels must also be evenly divided between the two fields in the vertical direction:

$$nOvsize = nEvsize = \frac{1}{2} 500 \text{ Pixel} = 250$$

(c) Positioning

It is useful to center the image.

In the X-direction, only 700 of the 720 pixels are displayed in the window. Thus, a border of $720 - 700 = 20$ pixels exists. The 20 pixels are evenly distributed on both sides, i.e. 10 pixels on the right and 10 pixels on the left. $hpos$ is the size of the left-hand border, therefore $hpos = 10$.

$$nOhpos = nEhpos = 10 \text{ (note this has to be an even value)}$$

Correspondingly in the Y-direction: $(288 - 250):2 = 19$;

$$nOvpos = nEvpos = 18 \text{ (note this has to be an even value)}$$

The parameter $nInterlaced$ should be set to 1 so that the images are automatically interlaced into one common memory location.

After size and resolution of the image are defined, the data format has to be selected. The parameter *Colformat* describes the format, a pixel is to be stored in the memory of the computer and how many bytes are used to represent one pixel.

The format is determined by the application. In general, three formats can be distinguished, which again are subdivided in different formats. *Figure 59* shows how pixels can be stored in the memory for different formats.

- **RGB** : The information for brightness of the three color channels red, green and blue is stored individually for each color. This is the common way of processing and handling color information in many applications.
 - *RGB32*: 32-bit, a double word per pixel is used to store RGB color format. The lowest byte of each double word contains the information for the blue color (8-bit width), the second byte the green and the third byte the red color information. The highest byte contains no information and is used only to obtain a double word format for one pixel: After each double word a new pixel begins (see *Figure 59* where related information have same hatching).
The alignment on double word borders has the advantage, that fast access commands can be used.
The number of colors is 16 million ($2^{3 \cdot 8} = 16,777,216$).
 - *RGB24* delivers the same information as RGB32, but does not contain the stuffing byte. The image has the same color depth but occupies a smaller portion of the memory.

- *RGB16* has a reduced resolution of the color. This format requires five bit for the blue and red channel, and the green channel has 6-bits. The color depth is 65.536 ($2^5 \cdot 2^5 \cdot 2^6 = 65536$). One pixel is represented by 16 bits = 1 word. In *Figure 59* the partitioning into three color channels of the word is depicted. The color information is arranged „left justified“. This means the lower bits are omitted, so that the color depth is reduced. The color depth of the green channel is twice of each of the two other channels.
- *RGB15*: Corresponds to the RGB16-system, except that all color channels have the same color depth (each 5 bit = 32 levels). Therefore we yield 32.818 colors. Altogether only 15 bits are necessary, so that the upper bit of a word is a stuffing bit and has the value 0 (see *Figure 59*).
- **YCrCb**: In this format grey values and color information is stored separately. The parameter Y describes the brightness of the pixel (the grey value) and the parameter tuple (Cr,Cb) provides the color information. (Cr,Cb) can be considered as a vector in the chromatic circle. The tone of the color corresponds to the angle of the pointer, the saturation is presented by the absolute value of the vector. This format is applied for streaming purposes. It is compact for the storage of images and also a good base for MPEG or JPEG compression.
- *YUY2*: This format corresponds to the format YCrCb 4:2:2. In one double word the information of two pixels is stored. Y0 and Y1 is the information for the brightness of two adjacent pixels, Cb0 and Cr0 presents the color information of the first pixel, which is used for both pixels. The color information of the second pixel is not used.

- *BtYUV* : corresponds to YCrCb 4:1:1. Four pixels share one color information. The arrangement of the information in the memory is shown in *Figure 59*. Three double words are logically combined to one unit and contain the information for eight pixels. This is the value for the brightness for each pixel (Y0..Y7) and the color information of the first (Cb0/Cr0) and fifth pixel (Cb4/Cr4).
- **Y :** In the grey scale format only the grey value, the value of the brightness of a pixel, is stored. The color information is not considered. This format is recommended if color is not to be used for the processing the image.
 - *Y8*: In the Y8-format the grey value of each pixel is stored in sequence as a value with 8-bit, so that one byte corresponds to one pixel.

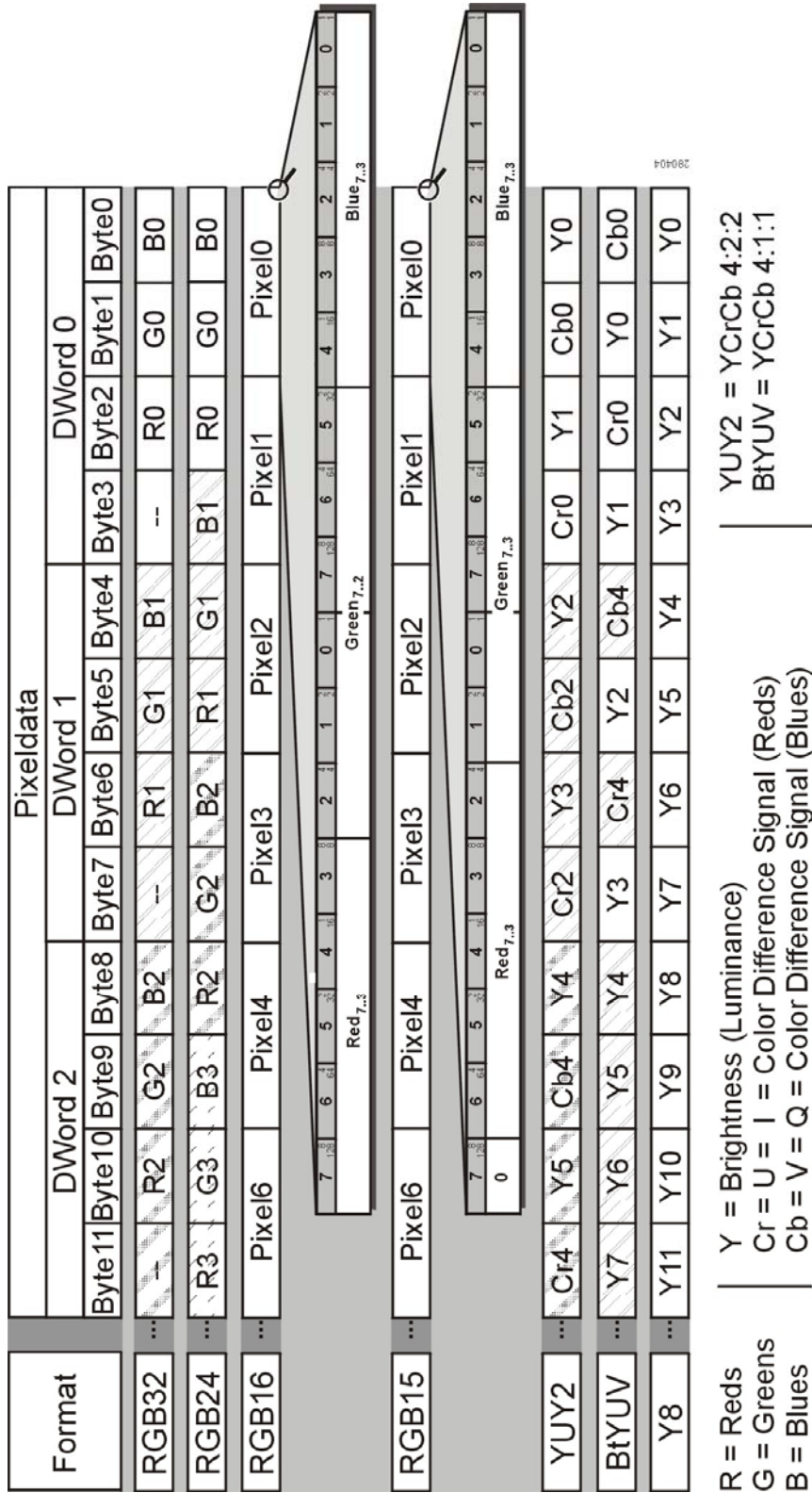


Figure 59: Color Format of the pciGrabber-4plus/express

Now the format of an image to be digitized, is exactly defined. The framegrabber must now be instructed where and how to store the data of the image in the memory.

The Windows driver reserves an image memory space, in which the image is placed.

How much memory will be used? This will be calculated from the size of the image (number of pixels) and the required number of bytes per pixel (color depth):

Memory requirements per field = $hsize \cdot vsize \cdot pixel\ size$ [byte]

The value *pixel size* can be depicted from *Table 14*.

For the format YUV2 and BtYUV it must be considered that 2 or 8 pixels are combined logically and that the resolution of the image is selected correspondingly. The value calculated for the required region of memory is valid for **one** field (*even* or *odd*). These values must be added if a frame is required

Format	<i>pixel size</i> [Byte]
RGB32	4
RGB24	3
RGB16, RGB15	2
YUY2	4 Byte per 2 pixel
BtYUV	12 Byte for 8 pixel
Y8	1

Table 14: Required Memory Space of One Pixel for the Different Modi

If whole frames are required, because the resolution should be more than 288 lines, the framegrabber can be instructed to store the frame in one single memory region. The framegrabber automatically will interlace the two fields. If this option is required, *nInterlaced* is set to 1.

Finally the type of image capture is described: With *nSingleShot* = 0 the framegrabber is instructed to capture continuously. That means, that after the start of the capture process, informations are stored in the memory in real time (up to 50 fields per sec.). In field mode (*nInterlaced* = 0) the information is stored to one memory region (20 ms), then to the other region (another 20 ms) alternatingly. This means, that each field memory region is not accessed by the framegrabber at least for 20 ms.

For frame mode (*nInterlaced* = 1) the framegrabber stores continuously to the common memory, 20 ms the odd and then 20 ms the even lines.

During the processing of the image data, it might be disturbing that the framegrabber is writing new data to the same region, and a mismatch might occur for fast moving objects. In this case a stop-and-go operation is recommended, or capturing the frames to separate memory regions which are processed alternatingly (*nInterlace=0*)

nSingleShot = 1 has the effect that only one single capture takes place. Two fields are captured (one odd, one even) or one entire frame. The user can capture the images and with repeated starts new data are stored in the memory. This mode of operation is recommended, if only occasionally images are captured and no real time application is required.

In any case, if continuous or single shot grabbing is used: ***Set_Image()*** configures, *how* the image is recorded. Grabbing is not started with this function but with the instruction *Start_Grabber()* (see description below).

↔ ③ Start Capture

void Start_Grabber(WORD nDevNo);

The function *Start_Grabber()* starts capturing with the device specified by *nDevNo*. The result will be digitization with the beginning of the next available image. If continuous mode was selected, then one image after the other will be captured until the instruction *Stop_Grabber()* is called. For single shot operation digitization is finished after one complete image.

When does the first digitization take place?

The driver handles whole frames. Always even/odd image combinations are evaluated. The timing will thus depend on the desired field and the field applied at the video input at start up time. We have to distinguish the following cases:

(1) An even-field is requested for capturing

a) At the input an even-field is in progress

Since the image just in progress can not be captured anymore (the beginning is missing), the rest of the even field and the next odd field will pass before capture will start. So the next complete even field will be digitized.

The delay from the start instruction to the beginning of the capture process will be less than 40 ms.

b) At the input an odd field is in progress

The even field following the odd field will be digitized.
Maximum delay time: 20 ms.

(2) An odd-field is requested for capturing

a) At the input an even field is in progress

Since the driver evaluates even fields first, the rest of the incomplete even field and the following odd field will pass,

until the framegrabber will synchronize with the next even field.

Now the starting odd field will be digitized. The maximum delay will be less than 60 ms.

b) At the input an odd field is in progress

First the incomplete odd field will pass and then the framegrabber will be synchronized with the following even field, and will start the subsequent odd field. The maximum delay will be less than 40 ms.

⇔ **③ Stop Capture**

void Stop_Grabber(WORD nDevNo);

This routine *Stop_Grabber* aborts the capture process. The transfer of data will be cancelled immediately and the image might be incomplete.

Caution:

Stop_Grabber must be called also for capturing in single shot mode (*nSingleShot* = 1 in *Set_Image*) when operation is stopped automatically. The Grabber is locked (paused), until *Stop_Grabber* is called.

Thereafter a new single shot can be initiated with *Start_Grabber* .

↔ ③ Get Capture Status

WORD Data_Present(WORD nDevNo);

return value: shows status of capture process
(4-bit - values 0 - 15)

The function *Data_Present* indicates if an even or odd image is stored in the memory.

In the separate bits of the return value the status for continuous or single shot operation of the digitization is coded (see Figure 60).

Bit 0 and 2 indicate, that an even image was captured, bit 1 and 3 represent an odd image. Bits 0 and 1 toggle their state with each beginning of an even or odd image resp.. For continuous capturing, this indicates when the content of the corresponding memory region is written completely with a new image's data. Each alteration of the status (0 to 1 and 1 to 0) indicates, that a new field was captured.

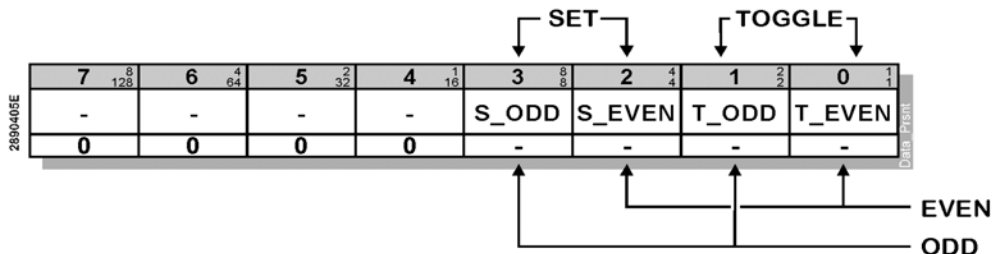


Figure 60: Return Values of Data_Present

Bits 2 and 3 are set to 1, as soon as an even or odd image was completely captured. Those bits are used, in case only one image is to be captured (single-shot, defined by *Set_Image*). The bits remain set until a new capture process is started.

Caution:

Do not call the status too often during capture, since each inquiry will occupy the PCI-bus, which might interfere with the data transfer of the grabber. It is strongly recommended to insert delay times between inquiries, in order to avoid slowing down digitization.

Please pay attention to evaluate the correct bits, which correspond to the actual mode, since otherwise the program might access the data during the wrong time interval.

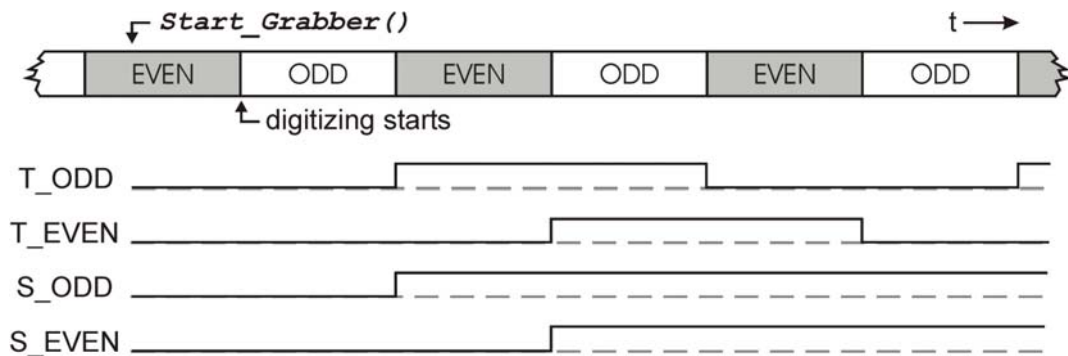


Figure 61: Timing Diagram of the Return Parameters of Data_Present()

3 Reading image data

DWORD GetPictureBufferAddress (WORD nDevNo, DWORD dwBitsSize) ;

dwBitsSize: size of the memory region used for the image

return value: start address of the memory region

In order to read the digitized data from the memory region controlled by the device driver, the user application must know the start address of the memory region.

The function *GetPictureBufferAddress* returns this address to the user application, at which the region of the memory starts, defined before by the driver. In this memory region the framegrabber stores the data of the digitized image with the following structure:

(a) Only even- or only odd-fields are captured

(the parameters of the dimension of the other field are zero)

⇒ The digitized field is placed at the beginning of the memory region of the image.

(b) Even- and odd-fields are digitized

(nInterlaced=0)

⇒ The even-field is placed at the beginning of the memory region. Adjacent to the even fields, the odd field is positioned.

The start address of the odd field is:

Odd Field Start Address = Start Address + (nEhsize · nEvsize · pixel size)

(c) Frames are digitized in interlace-modus

(nInterlaced=1)

⇒ The digitized frame is composed of interlaced fields and begins at the start address of the memory region of the image.

The parameter *dwBitsSize* defines the whole size of the required memory region of the image. This is calculated from the number of pixels and the number of pixels per Byte: $hsize \cdot vsize \cdot \text{pixel size}$.

Please note that *dwBitsSize* is the size of the **entire** memory region required for the image. If even- and odd-fields are digitized separately, you have to add the memory region for both fields. More information on calculating memory requirements can be found at the description of the *SetImage()*-function.

Caution:

When using the **Windows'95** driver, all framegrabbers in the system use the same image buffer. For this reason, two grabber cannot simultaneously digitize images when the standard routine for the driver is being used. If both Grabbers are in the digitization process, then image data stored in the buffer will be over written. This limitation does not exist for drivers under all the other operating systems.

   **Activate Interrupt**

**DWORD ActivateFieldInterrupt (WORD nDevNo,
HANDLE *hEvent);**

HANDLE: pointer to an event-handler

Monitoring the progress of image capturing can also be done using the hardware-interrupt of the framegrabber. An interrupt can be generated at the end of the capture of every field. To do so, an event is defined that the framegrabber triggers with the appearance of the interrupt (a field has been captured). Waiting for this event is a method to monitor the digitization progress without polling the status flags (see *DataPresent()*) and allows fast reaction.

The function *ActivateFieldInterrupt()* sets up the interrupt configuration of the grabber card and activates the hardware interrupt. It creates the event and returns a pointer (handle) to this event. You can monitor this event by using the Windows API-function *WaitForSingleObject()*. Please note, that the event has to be reset by calling the function *ResetEvent()*. Otherwise, new events can not be signaled.

Important:

- The event is signaled every time the capturing of a field is finished. This happens only, if the capture has been started by *Start_Grabber()*. After the occurrence of the field's end, the signal remains active until it is reset by calling *ResetEvent()*. This enables the application software to determine even some time later, whether the event had happened or not. To reset the event, the function *ResetEvent()* must be called. Otherwise, the next call of *WaitForSingleObject()* would return immediately, because the signal is still active. *ActivateFieldInterrupt()* should therefore be called only when the grabber is not running. It is recommended to call *ResetEvent()* after that, to reset any older active signals, which might have been created in the meantime.
- To determine which field had been digitized, the *DataPresent()* – function should be called immediately after the event had been signaled. In case of continuous capturing this is done by evaluating the state of the T_EVEN / T_ODD – flags. Care must be taken to read out this flags immediately, because their state will only persist for 20 ms after the end of the field. (This is, because 20 ms later, the next field digitization is completed and therefore the flags are updated).
Hint: An exclusive-or – combination of T_EVEN and T_ODD will result the parity of the field (1=ODD, 0=EVEN):
field_ready := T_EVEN ^ T_ODD ;
- The event is created by the grabber driver with a very short time delay. Please notice, that the delivery to the application by the operation system might last a noticeable (and non-constant) time.

⇔ **③ Checking for a video signal at the input**

WORD Get_Signal_Status(WORD nDevNo);

return value: 0 = no video signal at input
1 = undefined
2 = video signal present
3 = video signal present and line locked

With the help of this function it can be determined, if a camera is connected to the selected channel. In addition the quality of the signal can be evaluated.

The return value 0 indicates, that no source is connected to the selected channel. If no synchronizing pulses are detected for 31 lines, it is assumed, that no source is applied.

If the return value equals 2, a video source is applied. Consider that interference at the input might cause a wrong indication.

The return value 3 indicates a stable video signal. This value is generated if the horizontal synchronizing pulse is found within ± 1 clock of the expected position. This must be given for 32 consecutive lines. Vice versa this „HLOCK“ status will not be indicated, if this criteria is not fulfilled for 32 consecutive lines.

Therefore this indication will be very reliable. Some image sources such as a video tape recorders tend to not having a stable timing. For such devices it might be possible, that the Return value 3 will never be indicated. Nevertheless digitization will be faultless thanks to the ULTRALOCK™ synchronization system of the framegrabber.

↔ ③ ☆ Counting the Digitized Images

With the *pciGrabber-4plus/express*, it is possible to count the number of digitized images. The following two functions are used for this purpose:

These functions are not compatible with the *pciGrabber-4*.

BYTE Get_CaptureCounter (WORD nDevNo)

return value: Number of grabbed fields modulo 256

The function returns the number of captured fields. The result is a byte value and the counter resets from 255 to 0.

void Reset_CaptureCounter (WORD nDevNo)

Calling this routine sets the field counter back to zero.

⇔ ④ **Brightness Adjustment**

void Set_Brightness(WORD nDevNo, short nBright);

nBright: brightness of the image (-128..127)

This function specifies the value for the brightness in the video processor. The value is a constant added to the brightness of a single pixel. This brightness can be varied from -50 % to +49.6 %. One LSB corresponds to a change in brightness of 0.39 %:

$$\text{nBright} = \text{brightness}[\%] \cdot 2.5601 [1/\%]$$

⇔ ④ **Reading the brightness setting**

short Get_Brightness(WORD nDevNo);

return value: Content of the register holding the value for brightness in the video processor.

With this function the actual brightness setting of the video processor can be read back.

⇔ ④ **Contrast Adjustment**

void Set_Contrast(WORD nDevNo, WORD nContr);

nContr: contrast (0..511)

This function specifies the contrast of the image. The contrast is a constant factor which is multiplied in the video processor with the brightness of the pixel. The factor has a range of 0 % to 236.57 % :

$$\text{nContr} = \text{contrast} [\%] \cdot 2.1598 [1/\%]$$

↔ ④ Reading the contrast setting

WORD Get_Contrast(WORD nDevNo);

return value: actual contrast value

With this function the actual value of the contrast register of the video processor can be read. The returned value is an integer number.

↔ ④ Color Saturation Adjustment

**void Set_Saturation(WORD nDevNo,WORD nSat_U,
WORD nSat_V);**

nSat_U: Saturation of the U-color portion (0..511, Default = 254)

nSat_V: Saturation of the V-color portion (0..511, Default = 180)

This function allows the separate setting of the color saturation for the U- and V-color portion. With this parameters the amplification can be separately regulated for both color portions. Usually the ratio between the values *nSat_U* and *nSat_V* is equal. The control of the difference between the U- and V- portion allows the elimination of color faults (which might stem from unbalanced color cameras). The resulting effect will be a change of the color of the image.

$nSat_U = \text{U-saturation [\%]} \cdot 2.5400 [1/\%] ; 0\% \dots 201.18 \%$

$nSat_V = \text{V-saturation [\%]} \cdot 2.1396 [1/\%] ; 0\% \dots 238.83 \%$

↔ ④ Reading the Color Saturation Settings

WORD Get_Sat_U(WORD nDevNo);

WORD Get_Sat_V(WORD nDevNo);

return value: value of the actual U- or V-color saturation

This function provides the content of the registers for the color saturation.

⇔ **4 Correction of the hue (NTSC only)**

void Set_Hue(WORD nDevNo, short nHue);

nHue: hue, phase position of the color signal (-128..127)

With this function the hue for the digitization of NTSC- color images can be controlled, which is accomplished by changing the phase shift of the color signal. For PAL this value is insignificant since phase errors are automatically compensated.

One LSB corresponds to a correction of the phase angle of 0.7° , therefore the color signal can be varied in the range of -89.3° to $+90^\circ$

This value must be set to 0 to ensure proper functioning of the color decoder when using PAL sources.

⇔ **4 Reading the Hue Setting**

short Get_Hue(WORD nDevNo);

return value: value of the phase position of the color signal

This function provides the hue settings.

↔ 4 De/activation of the luma-low-pass-filter**void Set_LDec(WORD nDevNo, WORD nOn, WORD nHFilt);**

nOn: 1 = Luma decimation on
0 = Luma decimation off

nHFilt: 0 = automatic filter selection
1 = CIF filter
2 = QCIF filter
3 = ICON filter

For small image sizes, a higher quality is achieved, if the resolution of the input signal is reduced (so that sharpness of the image is adjusted to the resolution of the captured image). For this reason, an optional low-pass-filter can be inserted into the luma path.

With the parameter *nHFilt* the filter is adapted to the size of the image in the following way:

The function *automatic filter selection* adapts the filter setting to the size of the image. (see also *Set_image*). In addition the filter can be adjusted manually to one of the standard formats CIF (= 1/2 frame size), QCIF (1/4 frame size) and ICON (1/8 frame size).
Default: Luma-low-pass is turned off.

↔ 4 De-/Activation of the test image**void Set_ColorBars(WORD nDevNo, WORD nColorBars);**

nColorBars: 0 = test image off
1 = test image on

This function controls the activation of a test image. The test image contains vertical colored bars. The test image is independent of an input signal. In order to see the whole image the size of the image should have CIF-format or more.

⇔ ④ Adjustment of the Output Range

```
void LumaControl(WORD nDevNo,  
                WORD nRange,  
                WORD nCore);
```

nRange: 0 = luma range 16 - 253
 1 = luma range 0 - 255
nCore: 0 = 0; all brightness values are output
 1 = 8; all brightness values <= 8 are set to 0
 2 = 16; all brightness values <= 16 are set to 0
 3 = 32; all brightness values <= 32 are set to 0

With this function the output format of the brightness and color values can be adapted to the application

The parameter *nRange* determines the range of values for the brightness (permissible grey values).

- *nRange*=0 corresponds to the standard range of values specified in CCIR 601. The range of values for the brightness is restricted to the values 16 to 253 where Y=16 corresponds to black. The range of values for colors is 2...253 with Cr/Cb=128 as zero (signed).
- *nRange*=1 allows the utilization of the entire range, that is for Y the range 0...255 with 0=black, the chroma range is defined as for *nRange*=0.

↔ ⑤ Reading/Writing data via the option port (GPIO-Port)

12 I/O pins are available on the pin header row option port. These pins can be controlled with the following three functions:

void Set_GPIO_Direction (WORD nDevNo, WORD nDirection);

void Set_GPIO_Data (WORD nDevNo, WORD nData);

WORD Get_GPIO_Data (WORD nDevNo);

nDirection: can have values between 0 and 4095
(direction control of the port)

nData: Data, which should be output via the option port

return value: Get_GPIO_Data(): Data read from the option port

The *pciGrabber-4plus/express* has an option port with 12 I/O pins, which can be used separately to read or write digital signals. With the GPIO functions the option port is controlled. You can configure which pins will act as input or output, set output pins to low or high level and read the levels that are applied to the input pins.

With the help of the function **Set_GPIO_Direction** each pin of the port can be configured as input or output. For this purpose the lower 12-bits of the parameter *nDirection* are used. If a bit is set to 1, the corresponding port is configured as output. A 0 results in a configuration as input.

Caution:

Please pay attention, that a pin of the port may only switched to output, when no external signal is applied to the pin. Otherwise permanent damage can be caused to pin circuitry.

All pins are configured as inputs when starting the computer. Please note that the pins are high ohm and therefore the logic level is not defined. Hardware precautions (i.e. pull-up resistors) must be taken in order to determine the behavior of components controlled by the pins during the start-up of the computer until the GPIO port has been configured.

With **Set_GPIO_Data** a „High“ = 1 or „Low“ = 0 level can be set to each port pin which is configured as an output. nData is a 12-bit value, whereas to each bit one pin is assigned. The setting is only effective for those pins, which are configured as output.

The function **Get_GPIO_Data** reads the data from the port pins, which had been selected as input and returns a 12-bit value. The return value for the pins configured as output, will be the actual setting.

🔍 ⑤ ☆ DIP Switch**WORD Get_DIPSwitches(WORD nDevNo)**

return value: status of the DIP-Switches in the lower 4Bit

With this function the status of DIP-Switches can be read.

Note:

The DIP Switches are only supported on pciGrabber with -RS6 option installed.

🔍 ⑤ ☆ Relais**void Set_Relais(WORD nDevNo, WORD nData)**

nData: Number of relay to be set (value 1...4)

void Reset_Relais(WORD nDevNo, WORD nData)

nData: Number of Relay to reset (value 1...4)

With this function you can set or reset the relays installed on grabber cards with relay option.

A relay is activated (switched on) by calling *Set_Relais* with the corresponding number of the relay.

Calling *Reset_Relais* switches the designated relay to off-state.

Note:

The Relays are only supported on pciGrabber with an RS6 extension.

⇔ ⑤ ☆ Transmitting Data via the I²C Interface

This functions allow to read and write data from / to to devices connected to the I²C interface of the framegrabber card.

Caution:

The I²C-EEPROM, found on the framegrabber card, is protected against accidental writing. Therefore, access to the device address space 0xA0 to 0xA3 is not allowed.

In order to obtain access to the internal EEPROM memory space, please use the appropriate special functions.

void I2C_Set_BR_Mode (WORD nDevNo, BYTE bMode)

bMode baudrate
 pciGrabber-4*express*,
 pciGrabber-4*plus*: 0 = 99,2 kHz, 1 = 396,8 kHz
 pciGrabber-4: 0 = 33 kHz, 1 = 290 kHz

This function determines the baud rate for transmission on the I²C bus. A lower or a higher transmission rate can be selected.

BYTE I2C_ReadByte (WORD nDevNo, BYTE bChipAddress, BYTE bSubAddress, BYTE *bByteRead)

bChipAddress: Device address for the I²C device on the bus
bSubAddress: Internal memory address for the I²C device
*bByteRead: Pointer points to a Byte variable. Result is written into the Byte variable.

return value: SUCCESS, NOACK, INVALID_ADDRESS

I2C_ReadByte is used to read a byte from the memory space of an I²C device. The result is given in a variable of type *byte* which has to be defined before.

The function returns an error code as a return value. `NOACK` indicates that no I²C device has been responded under the specified device address. `INVALID_ADDRESS` indicates an unsuccessful attempt to access the protected area of the EEPROM mounted on the framegrabber.

BYTE I2C_WriteByte (WORD nDevNo, BYTE b ChipAddress, BYTE bSubAddress, BYTE bData)

bChipAddress: Device address of the I²C device on the bus

bSubAddress: Internal memory address of the I²C device

bData: Byte written into the specified address

return value: `SUCCESS`, `NOACK`, `INVALID_ADDRESS`,
`WRITE_FAILED`

Writes a Byte into the memory space of a specified I²C device.
The function returns an error code as a return value.

Note:

Under certain circumstances some devices – as for example EEPROMs – need longer times to process a write command than the I²C-bus cycle lasts. If several write commands follow one another too fast, these devices return an error message (`NOACK`) as they are not ready to receive the new data.

To avoid this, after writing to the device it is recommended to perform a read command and verify that the value read back is equal to the written value. If this is the case, the internal write operation is finished successfully and next write operation to this device can be performed.

Please also refer to the documentation of the specific I²C device.

⇔ ⑤ ☆ Using Internal EEPROM

The pciGrabber-4plus/express has an internal non-volatile memory. This memory is intended for the storage of parameters. A total of 252 Bytes is available to the user.

Note:

Since the predecessor models to the pciGrabber-4plus/express do not include internal memory, the following functions are not compatible.

BYTE I2C_ReadEEProm (WORD nDevNo, BYTE bSubAddress, BYTE *bByteRead)

bSubAddress: Memory address to be read (0x00 ... 0xFB)

*bByteRead: Pointer points a the byte variable. The result is written to the byte variable.

return value: error code = SUCCESS, NOACK

Reads a byte value from the EEPROM. When calling the function the memory address that is to be read must be specified. The result is given in a variable of type byte which must have been defined before.

The function returns an error code as a return value (see *I2C_ReadByte*).

BYTE I2C_WriteEEProm (WORD nDevNo, BYTE bSubAddress, BYTE bData)

bSubAddress: memory address written to (0x00...0xFB)

bData: Data byte that is written

return value: Error code = SUCCESS, NOACK, WRITE_FAILED

Writes a byte value to the internal EEPROM. The desired memory address and the byte value to be written to it have to be specified. The function returns an error code (see *I2C_WriteByte*).

Note:

The life time of the internal EEPROM memory is 1 million write accesses. The number of read accesses is unlimited.

↔ 5 ☆ Controlling the I/O Pin

The *pciGrabber-4plus/express* offers an externally available, transistor driven I/O pin. It can be used either as input or output. The following functions are used to control the I/O pin:

Note: This function is not available for the *pciGrabber-4*.

void Set_Ext_IO (WORD nDevNo, BYTE bData)

bData: Control value, 0 = CLOSE, 1 = HI-Z

This function controls the output switch of the port pin. The output pin's transistor operates like a switch connected to ground. If the control value is set to 0, the switch is closed and current will be conducted from the I/O pin to ground.

If the control value is set to 1, then the transistor is disabled (high impedance / switch open).

For more information on the technical specifications for the I/O output, please *refer to section 3.5.4 and 4.5.4*.

Note:

When starting the computer, the transistor is disabled (high impedance). Connecting an external pull-up resistor creates a logic „1“ signal.

A connected load, i.e. a relay, is not actuated (switched off).

BYTE Read_Ext_IO (WORD nDevNo)

return value: State of I/O pin

This function allows the state of the I/O pin to be read. This enables the pin to be used as input.

If the voltage connected to the I/O pin is in the range of „logic 0“, then the function returns a value of 0. If the voltage is in the range of „logic 1“, then the function returns a value of 1. For the appropriate voltage ranges, please *refer to section 3.5.4 and 4.5.4*.

Caution:

In order to use the I/O pin as input, the transistor must be disabled (high impedance).

↔ 5 Direct access to the video processor's registers

**WORD Read_Local_DWord(WORD nDevNo,
WORD nRegister_Number,
DWORD *lContent);**

nRegister_Number: number of the register
lContent: contents of the registers

**WORD Write_Local_DWord(WORD nDevNo,
WORD nRegister_Number,
DWORD lContent);**

nRegister_Number: Number of the register
lContent: data to be written to the register

Almost all the functions of the *pciGrabber-4plus/express* can be controlled with the routines of the driver. We expressly recommend the use of the standard functions.

In the case the user would like to affect directly the registers of the *pciGrabber*, these two functions are available. With these functions it is possible to read or write all registers of the video processor. Should you need further information about the registers of the *pciGrabber*, please contact the PHYTEC support team.

Attention!

Some registers of the *pciGrabber-4plus/express* must be set in the present device configuration with certain values to guarantee the function of the card.

Check hence the exact meaning of the register before making changes. PHYTEC cannot assume liability for damages which possibly originate from manipulation of the registers.

7.3 Driver for DOS Applications

The PCI4GRAB driver library, can be found on the CD's directory, under ***PCIGRAB4\DRIVER\DOS\DRIVER***. The *pciGrabber-4plus/express* can be adapted to DOS using this library.

Caution:

Please note that the DOS driver is not longer supported.
All information given in this section is for reference only.
The use of the DOS driver is not recommended for new applications.

7.3.1 Premises

Under the DOS operating system, it is required, that for the operation of the *pciGrabber-4plus/express*, the whole physical address area of the PC can be addressed linearly. This is necessary, because the PCI-BIOS configures the register area of the framegrabber in the memory just according to its own rules without any influence by the user. Usually high addresses above the RAM region are utilized, which require special addressing mechanisms.

Caution:

In order to utilize the DOS-driver, the DOS-Extender ***DOS/4GW*** has to be installed. Programs, which drive the *pciGrabber-4plus/express* under DOS via the driver routines, have to be designed in such a way, that they can operate with ***DOS/4GW***.

DOS/4GW from Rational Systems is a DOS-Extender, which provides protected-mode-access. Many compilers support the application of ***DOS/4GW*** and allow combining it with EXE-files of the user program.

Caution:

Some DOS-systemprograms may cause problems working together with ***DOS/4GW***. Especially the utilization of ***EMM386.EXE*** might be problematical. Therefore we recommend not to use ***DOS/4GW*** and ***EMM386.EXE*** simultaneously.

7.3.2 Development Platform

The DOS driver software is object-oriented according to C++. It is possible to link the driver also to programs, which are written in a different programming language. In this case you have to consider the corresponding procedures to call these programs.

The condition for using this library is, that the compiler supports 32-bit code applications, since this library of the *pciGrabber-4plus/express* is a 32-bit library.

For the memory model you have to select *32-bit-Flat*. We recommend the setting of the option '*80386 Register Based Calling*'.

For the programming of the driver, the compiler Watcom C/C++ from Powersoft version 10.6 was used. This compiler allows the 32-bit programming and the linking of *DOS/4GW*.

The driver is shipped as a library (*.LIB) - file and can be linked with all compilers, which support the used format. The header files (*.H) are included in the user program. They provide all declarations of the different functions, which are provided from the library.

Before the start of the user application the *DOS/4GW*-extender must be loaded, if the compiler does not automatically link the extender and thus triggers the start automatically. *DOS/4GW* allows the DPMI-access, which is required for the functioning of the driver routines.

7.3.3 Functions of the DOS Driver *PCI4GRAB*

The `PCI_GRABBER4` class contains all of the functions for initialization and configuration, and can be found in the library `PCI4GRAB.LIB`. These functions also control the framegrabber events.

With an object from this class, all Grabbers in the system are controlled.

For easier identification, all of the functions have been divided into five groups. The number of each group is shown in a black circle.

The functions are classified as follows:

❶ Routines for Initialization

This group of functions must be called once before using the framegrabber to ensure that the framegrabber operates correctly.

❷ Routines that configure the Grabber for capturing

Functions from this group configure the framegrabber to the connected image source. These functions also determine the appearance of the captured image in memory (image size, color format, etc.) The user should consider whether each function is needed, and which parameters are necessary. These functions may be called-up several times during the processing of a program (i.e. when the input channel should be switched or if the image size should be changed).

❸ Routines for Executing and Controlling the Capturing

These functions start the image digitization, monitor the Grabbing process and end digitization.

❹ Routines for Configuring Image Parameters

Functions from this group enable configuration of parameters, such as brightness, contrast, saturation, etc. These functions are not necessary, but can be called at any time to adapt the final image to user needs.

⑤ Routines for Controlling Grabber Card Options

This category includes functions that do not directly influence the grabbing process, but rather deal with the features of the framegrabber, i.e. I/O port, I²C interface, etc. These functions need only be called when a corresponding framegrabber feature is implemented.

Most of the functions are identical for the Windows and DOS driver versions. Although, depending on driver type, a few differences do exist. In order to simplify porting from programs, the following symbols are used:

↔ Calling the function is identical for DOS and Windows
Nevertheless, please note that the variable types can differ depending on the operating system.

☞ This function is specific to Windows

DOS This function is specific to DOS

Caution:

In all of the following descriptions for routines, the parameter `nDevNo` is used. This parameter identifies the desired `pciGrabber-4plus/express` when multiple Grabbers are installed in the system. The number of installed Grabbers can be determined by the function `Max_Device_Number()`.

Compatibility to the pciGrabber-4

The driver is downwards compatible with the pciGrabber-4 (VD-007 and VD-007-X1). Programs that have been written for the pciGrabber-4 also function with the pciGrabber-4*plus/express*.

In order to ensure operation of functions for the pciGrabber-4*plus/express*, new or altered functions have been created for the driver.

New applications that use these altered functions cannot operate with the pciGrabber-4.

Functions that are not compatible with the older driver version for the pciGrabber-4 are denoted with a star (☆).

If the new features of the pciGrabber-4*plus/express* are to be added to existing applications, please take note of the features denoted with a star (☆).

⑤ Evaluation of the Error Messages

short Get_Error(void);

return value: 0 = no error
 1 = device number not found
 2 = bad register number
 3 = initialization failed
 4 = Grabber not found
 5 = unknown parameter value

↔ **❶ Determination of the number of available pciGrabber-4plus/express**

unsigned short Max_Device_Number();

return value: number of the pciGrabber found.

↔ **❷ Initialization of the Grabber and driver activation after power up**

void Initialize(unsigned short nDevNo);

↔ **❸ Reading Information about the Grabber Cards installed**

**short Read_GrabberInfo (unsigned short nDevNo,
 unsigned short wInfoType)**

wInfoType: specifies the type of information requested

return value: information requested

↔ **❹ Get Grabber name as text string**

**short Read_OrderCode (unsigned short nDevNo,
 unsigned char* sCodeString,
 unsigned long dwSizeOfString)**

*sCodeString: pointer to a string variable (must be defined previously)
 (25 bytes min.). The function will write the grabber name
 into this string variable.

dwSizeOfString: size of the string array

return value: error code

⇔ ② Grabber setting to the color-system used

**void Set_Color_System (unsigned short nDevNo
 unsigned short nColSys);**

nColSys: code for color system

⇔ ② Recognition of the video format

short Get_Video_Status(unsigned short nDevNo);

return value: 0 = 525 lines format (NTSC / PAL-M)
 1 = 625 lines format (PAL / SECAM)

⇔ ② Configuring the Composite Mode (Composite Inputs)

void Set_Composite(unsigned short nDevNo);

⇔ ② ☆ Configuring the S-Video Mode

**unsigned char Set_S_VideoEx(unsigned short nDevNo,
 unsigned char input);**

input: MINIDIN = Mini DIN socket is the input
 COMBI = Combi plug (HD-DB-15 plug) is the input
 AUTO = automatic configuration

return value: MINIDIN = signal applied at the Mini DIN socket
 COMBI = signal applied at the Combi socket
 NO_SIGNAL = signal not found (AUTO mode)
 NOT_SUPPORTED = pciGrabber-4 and COMBI configura-
 tion

↔ ② ☆ Configuring the Input Channel

```
void Set_ChannelEx(unsigned short nDevNo,  
                  unsigned short nChannel);
```

nChannel: Input channel to be configured (1..9 / 1..3)

↔ ② Selection of the luma notch filter for black/white -operation

```
void Set_BW(unsigned short nDevNo, unsigned short nOn);
```

nOn: 0 = composite-signal at the input (activate the luma notch filter),
1 = b/w-signal at the input (deactivate the luma notch filter)

↔ ② De-/Activation of the Interlaced Mode

```
void Set_Interlace(unsigned short nDevNo,  
                  unsigned short nInterlace);
```

nInterlace: 0 = Non-Interlace
1 = Interlace

↔ ② De/Activation of the AGC

```
void Set_AGC(unsigned short nDevNo,  
             unsigned short nCAGC,  
             unsigned short nAGC,  
             unsigned short nCrush);
```

nCAGC: 0 = Chroma AGC off
1 = Chroma AGC on

nAGC: 0 = AGC on
1 = AGC off

nCrush: 0 = none-adaptive AGC
1 = adaptive AGC

⇔ ② **De/Activation of the color killer**

**void Set_CKill(unsigned short nDevNo,
 unsigned short nCKill);**

nCKill: 0 = color killer off
 1 = color killer on

⇔ ② **Dropping fields/frames in a video signal**

**void TemporalDect (unsigned short nDevNo,
 unsigned short nDecField,
 unsigned short nFldAlign,
 unsigned short nDecRat);**

nDecField: 0 = drop frame(s)
 1 = drop field(s)
nAlign: 0 = odd field will be dropped first
 1 = even field will be dropped first
nDecRat: number of the fields / frames to be dropped out of 50 (PAL)
 or 60 (NTSC)

② DOS Setting the size and scaling of the image

```
void Set_Image      (unsigned short nDevNo,  
                    unsigned short nOhpos,  
                    unsigned short nOvpos,  
                    unsigned short nOhsize,  
                    unsigned short nOvsize,  
                    unsigned short nOppl,  
                    unsigned short nOlines,  
                    unsigned short nOColformat,  
                    unsigned char *pOImgBuf,  
                    unsigned short nEhpos,  
                    unsigned short nEvpos,  
                    unsigned short nEhsize,  
                    unsigned short nEvsize,  
                    unsigned short nEppl,  
                    unsigned short nElines,  
                    unsigned short nEColformat,  
                    unsigned char *pEImgBuf,  
                    unsigned short nColSystem,  
                    unsigned short nInterlaced,  
                    unsigned short nSingleShot);
```

nOhpos, nOvpos: position of the upper left corner of the odd section (hpos = horizontal, vpos = vertical)
nOhsize: size of the odd-section in X-direction
nOvsize: size of the odd-section in Y-direction
nOppl: horizontal resolution of the odd field (ppl = pixel per line)
nOlines: vertical resolution (number of lines) of the odd field
nOColformat: color format: (RGB32, RGB24, RGB16, RGB15, Y8, YCrCb 4:2:2, YCrCb 4:1:1)
pOImgBuf: address of the odd-image memory

nEhpos, nEvpos: position of the upper left corner of the even picture section (hpos = horizontal, vpos = vertical)
nEhsize: horizontal resolution of the even field
nEvsize: size of the even section in Y-direction
nEppl: required size of the even-video picture in X-direction (pixel per line)
nElines: vertical resolution (number of lines) of the even field
nEColformat: color format: (RGB32, RGB24, RGB16, RGB15, Y8, YCrCb 4:2:2, YCrCb 4:1:1)
pEImgBuf: address of the even image memory

nColSystem: code for color system (see *Set_Color_System*)
nInterlace: 0 = Non-Interlace
 1 = Interlace
 2 = Field Aligned
nSingleShot : 0 = continuous capturing
 1 = **one** single image captured

The routine *Set_Image ()* defines the size, the position and scaling of the image sections delivered by the framegrabber separately for odd and even images. In addition, the data format in that the picture will be stored later in the memory, will be defined, and the address of this memory region is determined.

The settings for both fields can be established separately and the parameters have a letter prefix 'E' = even image or an 'O' = odd image. Parameter without those letters are valid for both fields.

Please be aware that the parameters for these functions can be differentiated between the DOS and Window's driver versions.

Under the DOS operating system, the user must reserve a space in the computer's RAM for storing images. A pointer that indicates the beginning of the reserved memory space must also be handed over.

The control of parameters for resolution scaling and positioning is equal to the procedure for Windows (refer to section 7.2.8).

Now the format of an image to be digitized, is exactly defined. The Grabber must now be instructed where and how to store the data of the image in the memory.

Therefore it is required to reserve a region in the main memory of sufficient size.

This is achieved by utilization of the well known instructions for the allocation of memory (for example *malloc(...)*).

How much memory will be used? This will be calculated from the size of the image (number of pixels) and the required number of Bytes per pixel (color resolution):

Memory requirements per field = $hsize \cdot vsize \cdot pixel\ size$ [Byte]

The value *pixel size* can be depicted from *Table 15*.

For the format YUV2 and BtYUV it must be considered, that 2 or 8 pixels are combined and that the resolution of the image is selected correspondingly. The value calculated for the required region of memory is valid for **one** field (*even or odd*).

Format	<i>pixel size</i> [Byte]
RGB32	4
RGB24	3
RGB16, RGB15	2
YUY2	4 Byte per 2 Pixel
BtYUV	12 Byte for 8 Pixel
Y8	1

Table 15: Memory Requirements for a Pixel in the Single Mode

If small formats are required, we recommend to apply separate fields. In this case you allocate two memory regions with two pointers providing the start addresses. They are transferred to the function by the parameters *pOImgBuf* and *pEImgBuf* for odd- and even-fields. The parameter *nInterlaced* is set to 0, in order to indicate, that the fields are stored in separate memory regions.

If whole frames are required, because the resolution should be more than 288 lines, the framegrabber can be instructed to store the frame in one single memory region. The framegrabber automatically will interlace the two fields. If this option is required, you have to set *nInterlaced* = 1.

In this case the start address of the frame in the memory is transferred by the parameter *pEImgBuf*.

Note:

This region must contain both fields, so that the region must be twice in size of one field. The odd parameter *pOImgBuf* is not used in the interlaced mode (*nInterlaced*=1).

If only one field (even or odd) should be digitized, the pointer of the field, which is not digitized is set to `NULL`. Therefore if only the even field is required you have to set *pOImgBuf* = `NULL`.

Finally the type of image recording, is described: With *nSingleShot* = 0 the framegrabber is instructed to digitize continuously. That means, that after the start continuously digitized information are stored in the memory in real time (50 fields per sec.). In field mode (*nInterlaced* = 0) the information is stored to one memory region (20 ms), then to the other region (another 20 ms) alternatingly.

This means, that each field memory region is at least not accessed from the framegrabber for at least 20 ms.

For frame mode ($nInterlaced = 1$) the framegrabber stores continuously to the common memory, 20 ms the odd and then 20 ms the even lines.

During the evaluation of the image, it might be disturbing that the framegrabber is writing new data to the same region, and a mismatch might occur for fast moving objects. In this case a stop and go operation is recommended. Or, the image data can be grabbed in two separate memory sections, which contain the even and the odd field and analyzed in an alternating way (synchronization in a 20 ms cycle for image actualization).

$nSingleShot = 1$ has the effect that only one digitization takes place. Two fields are captured (one odd, one even) or one entire frame.

The user can process the images and then with a new start command, new data are stored in the memory. This mode of operation is recommended, in case only occasionally images are digitized and no real time application is required.

In any case, if continuous or single shot grabbing is used: ***Set_Image()*** configures, *how* the image is recorded. Grabbing is not started with this function but with the instruction ***Start_Grabber()*** (*see description below*).

⇔ ③ **Start of Capture**

void Start_Grabber(unsigned short nDevNo);

⇔ ③ **Stop Capturing**

void Stop_Grabber(unsigned short nDevNo);

⇔ ③ **Read Capture Status**

short Data_Present(unsigned short nDevNo);

return value: shows status of digitization
(values 0 - 15 (4-bit))

DOS ③ ☆ **Interrupt Configuration**

**unsigned short Set_Interrupt (unsigned short nDevNo,
unsigned long nInterrupt);**

nInterrupt: OFF = no interrupts generated (default)
VSYNC = interrupt is set at the end of each field

return value: Error Status

The *pciGrabber-4plus/express* generates an interrupt after the end of a field is detected at the video input. During a digitization event, the interrupt can be used in order to determine if the process is finished without the usage of a polling procedure. After appearance of an interrupt the reason of the interrupt can be determined with the function *Data_Present()*

Note:

The Interrupt function should be used only by experienced users.

DOS ③ ☆ **Resetting the Interrupt Flag****unsigned short Reset_Interrupt (unsigned short nDevNo);**

return value: error status

After recognizing an interrupt, the interrupt status flag has to be reset. This is done by the interrupt service routine, by calling the function *Reset_Interrupt*.

↔ ③ ☆ **Counting Digitized Images**

These functions are not compatible with the pciGrabber-4

unsigned char Get_CaptureCounter (unsigned short nDevNo)

return value: Number of grabbed fields modulo 256

void Reset_CaptureCounter (unsigned short nDevNo)

Calling this routine sets the field counter back to zero.

↔ ③ **Checking for a video signal at the input****short Get_Signal_Status(unsigned short nDevNo);**

return value: 0 = no video signal at input
1 = undefined
2 = video signal present
3 = video signal present and line locked

↔ ④ **Brightness Adjustment****void Set_Brightness(unsigned short nDevNo,short nBright);**

nBright: brightness (-128..127)

⇔ **④ Reading the brightness setting**

short Get_Brightness(unsigned short nDevNo);

return value: Content of the register holding the value for brightness in the video processor.

⇔ **④ Contrast Adjustment**

**void Set_Contrast(unsigned short nDevNo,
 unsigned short nContr);**

nContr: contrast (0..511)

⇔ **④ Reading the contrast setting**

unsigned short Get_Contrast(unsigned short nDevNo);

return value: actual contrast value

⇔ **④ Color Saturation Adjustment**

**void Set_Saturation(unsigned short nDevNo,
 unsigned short nSat_U,
 unsignedshort nSat_V);**

nSat_U: Saturation of the U-color portion (0..511, Default = 254)

nSat_V: Saturation of the V-color portion (0..511, Default = 180)

⇔ **④ Reading the Color Saturation Settings**

unsigned short Get_Sat_U(unsigned short nDevNo);

unsigned short Get_Sat_V(unsigned short nDevNo);

return value: value of the actual U- or V-color saturation resp.

↔ ④ Correction of the Hue (NTSC only)

void Set_Hue(unsigned short nDevNo, short nHue);

nHue: hue, phase position of the color signal (-128..127)

↔ ④ Reading the hue settings

short Get_Hue(unsigned short nDevNo);

return value: value of the phase position of the color signal

↔ ④ De/activation of the luma-low-pass-filter

**void Set_LDec(unsigned short nDevNo,
 unsigned short nOn,
 unsigned short nHFilt);**

nOn: 1 = Luma decimation on
 0 = Luma decimation off

nHFilt: 0 = automatic filter selection
 1 = CIF filter
 2 = QCIF filter
 3 = ICON filter

↔ ④ De-/Activation of the test image

**void Set_ColorBars(unsigned short nDevNo,
 unsigned short nColorBars);**

nColorBars: 0 = test image off
 1 = test image on

⇔ **④ Adjustment of the range of values**

**void LumaControl(unsigned short nDevNo,
 unsigned short nRange,
 unsigned short nCore);**

nRange: 0 = luma range 16 - 253
 1 = luma range 0 - 255
nCore: 0 = 0 all brightness values are transmitted
 1 = 8 all brightness values <= 8 are interpreted as 0
 2 = 16 all brightness values <= 16 are interpreted as 0
 3 = 32 all brightness values <= 32 are interpreted as 0

⇔ **⑤ Reading/Writing data via the GPIO port**

The following three functions control the GPIO port (part of the user connector):

**void Set_GPIO_Direction(unsigned short nDevNo,
 unsigned short nDirection);**

nDirection: can have values between 0 and 4095

**void Set_GPIO_Data(unsigned short nDevNo,
 unsigned short nData);**

nData: Data, which should be output via the option port

unsigned short Get_GPIO_Data(unsigned short nDevNo);

return value: Data, which are read from the option port

↔ 5 ☆ Transmitting Data via the I²C Interface

These functions allow the user to read and write to devices that are connected to the I²C interface.

**void I2C_Set_BR_Mode (unsigned short nDevNo,
 unsigned char bMode)**

bMode Baud rate
 pciGrabber-4*express*,
 pciGrabber-4*plus*: 0 = 99,2 kHz, 1 = 396,8 kHz
 pciGrabber-4: 0 = 33 kHz, 1 = 290 kHz

**unsigned char I2C_ReadByte (unsigned char nDevNo,
 unsigned char bChipAddress,
 unsigned char bSubAddress,
 unsigned char *bByteRead)**

bChipAddress: Device address of the I²C device connected to the bus
bSubAddress: Internal memory address for the I²C device
*bByteRead: Pointer points to the Byte variable. The result
 is written in the Byte variable.

return value: SUCCESS, NOACK, INVALID_ADDRESS

**unsigned char I2C_WriteByte (unsigned short nDevNo,
 unsigned char bChipAddress,
 unsigned char bSubAddress,
 unsigned char bData)**

bChipAddress: Device address for the I²C device connected to the bus
bSubAddress: internal memory address for the I²C device
bData: Byte that is written to in the specified address

return Value: SUCCESS, NOACK, INVALID_ADDRESS,
 WRITE_FAILED

⇔ ⑤ ☆ Using the Internal EEPROM

The *pciGrabber-4plus/express* has an internal non-volatile memory available to the user for storing parameters.

The total amount of memory available to the user is 256 Bytes.

Note:

Since the predecessor to the *pciGrabber-4plus/express* does not have internal memory, the following functions are not compatible.

**unsigned char I2C_ReadEEProm (unsigned short nDevNo,
unsigned char bSubAddress,
unsigned char *bByteRead)**

bSubAddress: memory address to be read (0x00 ... 0xFF)

*bByteRead : Pointer points to a Byte variable. The result is written into the Byte variable.

return value: error code = SUCCESS, NOACK

**unsigned char I2C_WriteEEProm (unsigned short nDevNo,
unsigned char bSubAddress,
unsigned char bData)**

bSubAddress: Memory address that is to be written to (0x00...0xFF)

bData: Data Byte that is to be written

return value: error code = SUCCESS, NOACK, WRITE_FAILED

⇔ ⑤ ☆ Controlling the I/O Pin

The *pciGrabber-4plus/express* has an external, transistor driven I/O pin. Using control signals, this I/O pin is freely selectable as input or output. Both of the following functions can be used to control the I/O pin:

This function is not available for the *pciGrabber-4*.

void Set_Ext_IO (unsigned short nDevNo, unsigned char bData)

bData: Control value, 0 = CLOSE, 1 = HI-Z

unsigned char Read_Ext_IO (unsigned short nDevNo)

return value: logical level of the I/O-pin

↔ **5 Direct access to the video processor's registers**

The following functions allow direct access to the registers of the video processor.

**short Read_Local_DWord(unsigned short nDevNo,
 unsigned short nRegister_Number,
 unsigned long *lContent);**

nRegister_Number: number of the register
lContent: contents of the registers

**short Write_Local_DWord(unsigned short nDevNo,
 unsigned short nRegister_Number,
 unsigned long lContent);**

nRegister_Number: Number of the register
lContent: data to be written to the register

7.3.4 Program Example DOS

In this section a simple program is described running under DOS, to store an image in the main memory.

For a simple example we restrict us to a constant format of the image and digitize a single shot. The format of the image is 256 x 256 pixels. We assume, that only one *pciGrabber-4plus/express* is installed in the computer. If more grabbers are installed, only the first framegrabber is used.

At first an object „Grabber“ is created which provides the interface to the *pciGrabber-4plus/express*. The image should be stored in an array designated with `pEWert`. At this point it is necessary to determine the multitude of data for the required image. The dimension of the image is 256 x 256 pixels, and should be stored as RGB-color picture, using the RGB24-format. RGB24 requires 3 Byte per pixel. So we need $256 \cdot 256 \cdot 3$ Byte for one image.

The operation of the framegrabber starts with the identification, to check if a framegrabber is available. `Max_Device_Number()` delivers the number of framegrabber boards installed in the system. If the number = 0, no framegrabber board is available in the computer, and the program will be cancelled prematurely.

If one or several framegrabber cards are found, the framegrabber will be initialized with the function call `Initialize(1)`. The parameter 1 indicates that the first framegrabber should be initialized. If several framegrabber boards are available the function `Initialize()` must be executed for all framegrabber cards.

It is advisable to check after each call with `Get_Error()` if any errors occurred and if the instruction was successful. In order to keep the general view, this is not done in this example.

With the call `Set_Channel(1,1)` channel 1 is selected for the Grabber #1. Digitization will take place for this channel. The following delay is used to synchronize the framegrabber to the signal of the camera. The next function is quite complicated: With the routine `Set_Image(1,...)` the parameters of the image for the Grabber #1 are set. We want to digitize only one even field. Therefore the parameters for the odd field are not required and are set to zero (zero means, that the values in the video processor are not changed). It is important to set the pointer of the odd memory region to `NULL`, which is an indication to the framegrabber, that no odd image has to be produced.

For the parameters of the even image, first the position of the section of the image is defined. The coordinates of the upper left corner are (1.1), so that the section will be found also in the upper left corner of the TV-picture. (**Caution:** (0.0) will leave the values unchanged, which were set before.) As demanded in the problem, we now define the number of pixels required for the resulting image of 256 x 256. The field delivered from the source will be digitized with the complete size of the format with the ratio 4:3. Therefore the size of the image will be set to 344 x 258. A maximum of 360 x 288 would be possible, but 344 x 258 is more suitable because the whole height of the image will be visible in the section. Since the width of the image is different in the right region 88 pixels will be invisible.

With `RGB24` the color format for the even image is defined. Per pixel, three Byte are created, one for each red-, green- and blue value (see also the instruction `Set_Image()` and Figure 59).

As pointer to the image memory region `pEWert` will be provided, which will point to the beginning of the defined array.

For the determination of the video processor setting by the driver, the applied video system must be known. For our example we assume a PAL-camera as image source, and therefore `PAL_BDGHI` is set.

Since we use only one field, *Interlace* = 0 (both fields are stored in separate memories; in addition for our example the odd image is obsolete).

We select the single-shot mode, since only one image should be digitized and not a continuous series.

Now digitization can be started, which is done by invoking the function `Start_Grabber(1)`. During the following while-loop the program waits, until the digitized field will be stored completely in the memory. For this reason we call the function `Data_Present()` for Grabber #1. The Return value will be concatenate with `0x04 AND`, in order to mask the third bit (bit # 2). This is the *S_EVEN*-bit (see *Figure 61*). This bit will be set, after the even image is stored completely in the memory. Since only this information is interesting to us, the other status bits are masked out. The delay time in the loop prevents, that the status is permanently inquired, which might be disturbing for the data transfer on the PCI-Bus.

After the end of the digitization, the instruction `Stop_Grabber()` has to be given, so that the framegrabber is set to the none operative status. Otherwise the framegrabber remains in the standby position and no further digitization can take place.

The image is now found in the defined array and can be evaluated. In order to repeat recording of a new image with the same dimensions from the same channel, the program can be repeated starting from the function call `Start_Grabber()`.

```

#include <dos.h>
#include <malloc.h>
#include <stdio.h>
#include <stdlib.h>
#include "pci4grab.h"

PCI_GRABBER4 Grabber; // Object for the access to the Grabber
static unsigned char pEWert[256*256*3]; // Memory for EVEN-image

main ()
{
    if((Grabber.Max_Device_Number()==0)
    {
        printf("No pciGrabber-4plus was found!!") ;
        exit(1) ;
    }
    Grabber.Initialize(1) ; // First initialize Grabber!
    Grabber.Set_Channel(1,1) ; // Select input channel and
    delay(100); // Wait, until synchronized
    Grabber.Set_Image( 1, //Grabber-number
        0,0, //All ODD-parameters are...
        0,0, //...negligible and ...
        0,0, //...therefore...
        0, //...are set to 0 .
        NULL, //Pointer to ODD-image=NULL
        //=> no ODD-image
        1, 1, //upper left corner of the EVEN-
            image
        256,256, // Size of the image section
        344,258, // Size of the image
        RGB24, // Color information: RGB24-image
        pEWert, // Pointer to EVEN-image memory
        PAL_BDGHI, // Color system
        0, // Interlaced-modus off
        1); // Single shot on

    Grabber.Start_Grabber(1); // Start grabbing
    while(!(Grabber.Data_Present(1)&0x4))
        delay(10); // Wait until data are available
    Grabber.Stop_Grabber(1); // Stop grabbing
    // ** Image is stored in the memory and can be evaluated **}

```

8 Changes to the pciGrapper-4 and Compatibility

8.1 Changes between the pciGrabber-4 and pciGrabber-4plus

The *pciGrabber-4plus* is a successor to the *pciGrabber-4*. This means, by design, that these models possess downwards compatibility. That means:

- **Windows Applications:**

The *pciGrabber-4plus* can replace any *pciGrabber-4s* that have previously been installed in systems. Under the Windows operating system, there are no changes for existing application software.

Although, a new device driver must be installed for the *pciGrabber-4plus*.

The *pciGrabber-4plus* offers the same functionality as the *pciGrabber-4*. Additional functions for the *pciGrabber-4plus* cannot be used with older generation application software for the *pciGrabber-4*.

- **DOS Applications:**

Under DOS applications a separate device driver is not used, rather the driver is linked as a library for the compiler in user programs. In order to ensure that the *pciGrabber-4plus* operates with these programs, a new driver library must be linked. This means that the program must be newly compiled. There are no changes for user specific program code. The software interfaces for the driver library under the *pciGrabber-4plus* and the *pciGrabber-4* are also compatible.

- **Old Framegrabber and New Applications**

It is possible to install software that has been developed for the pciGrabber-4*plus* into a system populated with the pciGrabber-4 (VD-007) and subsequently use the new driver.

This is possible because the new driver supports older card versions. Although, please note, that many hardware applications are no longer available.

If functions or parameters are used that the older framegrabber does not support, the driver sends an error code to the user's program.

The functions *Read_GrabberInfo* and *Read_OrderCode* can be used to determine which Grabbers and features are available.

- **Hardware Compatibility**

All of the signals available on the pciGrabber-4 are also available at the same sockets and contacts on the pciGrabber-4*plus*. Connector cables previously used for the pciGrabber-4 can also be used for the pciGrabber-4*plus*.

For certain plug connectors, there are several additional pins that are supplied with signals that were not active with the pciGrabber-4. Please ensure, that any cables being used are not incorrectly connected to these pins.

The following table shows the pin assignments of the corresponding sockets. Newly connected signal pins are accented.

First HD-DB-15 (X1)		Second HD-DB-15 (X2)	
Pin	Function	Pin	Function
1	Composite Input 1	1	Composite Input 6
2	Composite Input 2	2	Composite Input 7
3	Composite Input 3	3	Composite Input 8
4	S-Video: Luma	4	S-Video: Chroma
5	Signal Ground	5	Signal Ground
6	Signal Ground	6	Signal Ground
7	Signal Ground	7	Signal Ground
8	Signal Ground	8	Signal Ground
9		9	I/O-Pin
10	Signal Ground	10	Pwr Supply Ground(-)
11	Signal Ground	11	Signal Ground
12	I²C Bus: SDA	12	I²C Bus: SDA
13	Composite Input 4	13	Composite Input 9
14	Composite Input 5	14	+12 V out (Camera supply)
15	I²C Bus: SCL	15	I²C Bus: SCL

bold = new signal pins

Table 16: Pin Assignment for the HD-DB-15 Sockets, Model VD-009

First HD-DB-15 (X1)		Second HD-DB-15 (X2)	
Pin	Function	Pin	Function
1		1	Composite Input 1
2		2	Composite Input 2
3		3	S-Video: Luma
4	S-Video: Luma	4	S-Video: Chroma
5	Signal Ground	5	Signal Ground
6	Signal Ground	6	Signal Ground
7	Signal Ground	7	Signal Ground
8	Signal Ground	8	Signal Ground
9		9	I/O-Pin
10	Signal Ground	10	Pwr Supply Ground(-)
11	Signal Ground	11	Signal Ground
12	I²C Bus: SDA	12	I²C Bus: SDA
13		13	Composite Input 3
14		14	+12 V out (Camera supply)
15	I²C Bus: SCL	15	I²C Bus: SCL

bold = new signal pins

Table 17 Pin Assignments for the HD-DB-15 Sockets, Model VD-009-X1

Option Port, X6					
Pin	Function	Pin	Function	Pin	Function
1	+5V out	8	I/O 6	15	I/O-Pin
2	I/O0	9	I/O 7	16	I/O Clk
3	I/O1	10	I/O 8	17	I ² C SCL
4	I/O2	11	I/O 9	18	I ² C SDA
5	I/O3	12	I/O 10	19	GND
6	I/O4	13	I/O 11	20	GND
7	I/O5	14	N.C.	bold = new function	

Table 18: Pin Assignment of the Option Port - Connectors (Both Models)

- **Power Supply for the Camera**

A +12 V DC power supply is available on the second HD-DB-15 socket in order to supply a connected camera with an operating power supply.

Previously, with the pciGrabber-4, the power supply was taken directly from the PCI bus. The maximum power consumption was limited to 400 mA.

Now the pciGrabber-4*plus* obtains power directly from the PC power supply via a floppy power supply plug. This offers the camera a total power consumption of 1.5 A supplied from the framegrabber.

For compatibility purposes, the pciGrabber-4*plus* can be configured as follows in order to replicate the pciGrabber-4's power supply (via the bus).

- Remove the <F2> fuse
- Install a 500 mA fuse (fast blow) at socket <F1>. This fuse is not included in shipment; PHYTEC order number: KF014.

Caution:

Use only a fuse with a maximum of 500 mA at this socket. Use of a higher rate can cause damage to the framegrabber or the PC and could cause a fire.

- **Changing the Old Driver Version (for V3.0)**

If software has been developed for the pciGrabber-4 that is still used in conjunction with older driver versions (before Version 3.0), please read the following notes:

- The Window's DLL was expanded with the function **GetVersionNumber**. If a non-valid function pointer is returned with **GetProcAddress**, then a newer version of GR4CDLL.DLL is required.

- **Set_Image Function:**

In previous versions, the *hpos* and *vpos* parameters in the Window's DLL were required to be set to 1 in order to place the cropped image in the upper left-hand corner of the digitized image. Now the parameters accept only even values, this means that *hpos* and *vpos* must be set to 0.

8.2 Changes between the *pciGrabber-4plus* and *pciGrabber-4express*

The *pciGrabber-4express* is a *pciGrabber-4plus* with an additional PCI Express-to-PCI-Bridge. That bridge connects the PCI videodecoder to the PCI express bus. Note that the variant VD-009-X1 of the *pciGrabber-4plus* is compatible to the *pciGrabber-4express*. The variant VD-009 is available only as a PCI device. Because of the bridge, the *pciGrabber-4express* is fully compatible to the *pciGrabber-4plus*.

The *pciGrabber-4express* has two BNC sockets instead of the upper HD-DB-15 socket. This allows the use of BNC cables to connect two video sources to the framegrabber card. On the first BNC socket is channel 1 available and on the lower BNC socket channel 2.

Caution:

Standby Mode is supported by the *pciGrabber-4express* only, if the 3.3V supply of the PCI Express-Bus will be present during standby mode.

9 Trouble-Shooting

- The color representation is reduced in the Windows-demo-program.
 - Check the configuration of the graphic card. In order to yield the full resolution of color of the *pciGrabber-4plus/express*, the graphic card must be set at least to 16 Mio. colors.
- Only a blue image is displayed.
 - The selected input is not connected to a video source. Usually a blue image is shown. Please check if the correct input channel is selected.
- The digitized image shows streaks and stripes
 - It might be a Moiré-effect caused by the color signal. Check if the luma notch filter is enabled.
 - The cable to the camera might be defect (check shielding).
 - A ground loop might be existent by an additional ground connection between PC and camera causing a mains hum. Check, if ground loops or power supply cause a mains hum .
- At the S-video input no image is digitized.
 - Did you connect the S-video input properly?
 - Are both S-Video inputs connected (only one's possible)?
 - Was the *pciGrabber-4plus/express* configured to the S-video operation?
 - Has an incorrect input socket been selected (Mini DIN / Combi)?
 - Has another channel been selected after switching over to S-Video with Set_Channel?

- For S-video operation the image is only black/white
 - Is the chroma-signal properly applied?
 - Is the framegrabber configured to S-video operation?
- For S-video operation the image is black/white with color disturbances.
 - Is the chroma-signal properly connected?
 - Is a S-video-source connected (not a composite)?
- The image is displayed incompletely/ or very slowly
 - Increase the delaytime between status inquiries. The PCI-bus might be blocked by too many calls concerning the status.
 - Was the resolution reduction concerning to time used?
- The image has no contrast / too bright / too dark
 - Check the setting of brightness, contrast etc.
 - If necessary activate the AGC
- The image skips or the fields are mismatched
 - The time delay during switching the channels was too short.
- The image appears without color / with the wrong format
 - Is the appropriate color system selected?
 - Was the framegrabber correctly initialized?
 - Was the delaytime after channel switching long enough?

- The computer stalls under DOS during the start of the program/ an error message appears
 - Another device driver or the memory management for example EMM386, might cause trouble. Remove the installation of those components or use other drivers.
- The pciGrabber-4express doesn't work properly after a standby mode of the Windows operating system
 - The pciGrabber-4express supports the standby mode under Windows only if the 3.3 volt power supply on the PCI Express bus will not be turned off. Otherwise, a restart must be performed.
- The lower edge of the image is missing/no image is digitized
 - The vertical image size was chosen too large.
 - It was not recognized, that starting from a line number higher than 288 lines (PAL) or 262 (NTSC), a whole frame must be used.
- The colors are not presented properly.
 - The values *hpos* and *hsize* must be even, in order that the color decoding works properly.
 - In the user program: You mixed up Cr/Cb .
 - Did you select the correct color format?
 - The chroma gain registers were changed.
Please pay attention, that for a correct Hue setting the U- and V-component must be identical in respect to their *percentage* value. But the values of the *registers* are different!

- During continuous grabbing the image jumps one line up/down.
- The parity of the field is neglected. Pay attention in order to demand always the same field. The memory region for the even and odd image should not be the same!

Note:

The mismatch is actually a half line, therefore the mismatch can not be compensated by displacing one line.

- The video source delivers no proper signal.
 - Switching channels between two cameras occurs too quickly
- During the display of frames diagonal lines/circles are interlaced or have unclear perimeter edges.
 - Even and odd fields are interchanged (only possible in case you use single fields).
 - After digitization an image, no further digitization is possible.
 - Before requesting new digitization, the last operation must be finished with the call of the function *Stop_Grabber()* .
 - For the DOS-demo program only a black/white image is displayed (the first two options in the menu are missing).
 - The DOS-program requires for the presentation of color pictures and for pictures with higher resolution, a graphic card equipped with a graphic processor ET4000 / ET6000 (Tseng Labs). The program example under DOS, does not take into consideration all the peculiarities of the various graphic cards. For the presentation with true colors for a resolution of 640 x 480 pixels the standard-VGA-Modus is not sufficient.

- In frame mode, using continuous monitoring, a shadow effect is recognized, despite the framegrabber digitizing two complementary fields in real time.
 - The effect is due to the slow speed of the host PC displaying the image. The update at the screen is not fast enough, even through the images are found digitized in the main memory.
- The supply voltage output for the camera is not functioning.
 - Check the mini fuse .
 - Is a cable from the PC's power supply connected to the plug connector on the framegrabber at X7? (small floppy power supply plug 5 V/GND/GND/12V)
 - Check the proper connection of the cable: The supply voltage is only available at the lower HD-DB15-socket. If the cable is connected to the upper plug, the power plug is connected to the video input 5.
- The displayed image is disturbed by horizontal stripes, which might show parts of the preceding image. For moving objects horizontal lag effect occur.
 - The framegrabber is not able to transfer the image data in real time via the PCI-bus, since other cards on the bus stress the bus too much, or the bus-configuration of the BIOS is not correct. Please check the configuration of the other PCI-cards and the configuration of the BIOS.

How is it possible to use several pciGrabber-4*plus/express* at the same time with Windows'95?

- Under Windows'95 all pciGrabber-4*plus/express* use the same memory portion of the main memory of the PC. So only one digitization process can be executed at the time. But during digitization of one framegrabber the other grabbers can respond and parameters can be written or read or be changed. This is an advantage for example if the channel selection should be changed and the time span could be used as lead time, so that the analogue signal is recognized correctly by the framegrabber. Under DOS the programmer can define a separate memory region for each framegrabber. Here parallel processing is possible. However it must be considered, that the transfer rate of the PCI-bus is not exceeded.

Under DOS or Windows 98/2000/NT/ME/XP/VISTA each framegrabber can utilize a separate memory space for image storage

Index

A

- accessories 13, 39
- analogue television technology 76
- Arithmetic Operations for**
- Static Images** 86
- Arithmetics*..... 86
- AUTO function
- Driver..... 132
- automatic channel search..... 72

B

- Basic parameters..... 82
- BNC plug..... 38, 64
- both fields 77

C

- color bars 86
- Color Bars* 86
- Color Meter* 85
- Color Mode*..... 72
- comb effect 78
- COMBI socket..... 33, 59
- Compatibility to the**
- pciGrabber-4**..... 182, 204
- Compatibility to the**
- pciGrabber-4**..... 123
- composite inputs 33, 59
- Composite Inputs..... 20, 46
- Composite Sources**..... 75
- connecting composite sources . 38,
 64
- connector cable 13
- Connectors**..... 18, 44
- Cross Hairs
- Blending In/Out 82

D

- Data Format** 15, 41
- delivery 39
- Demo program
- Image Settings 74
- Demo Program
- Channel Selection 74
- Image Resolution 75
- Image Selection 75
- Installation 68
- Operations..... 68
- Demo Programing
- Normalizing 87
- developing environments..... 108
- Device driver
- WIN 98/ME 65
- Win NT 4.0 66
- Win95 66
- DIP-Switch
- test (Demoprogramm)..... 91
- DLL..... 117
- Driver
- Configuring Composite Inputs
 130
- Configuring Input Channels 133
- Configuring the Interrupt.... 192
- Configuring the S-Video Mode
 131
- De-/Activating the Interlaced
 Mode 136
- Defining the Version Number
 for the DLL (Win) 125
- Grabber Name Read as a Clear
 Text String 128
- Programming I/O pins 175
- Re-Setting the Interrupt Flag 193
- Using Internal EEPROM 174

-
- Using the I2C Interface..... 172
 - Windows 98 / NT..... 116
 - driver installation 65
 - Drivers
 - Older Versions 208
 - E**
 - EEPROM 174
 - F**
 - Field Aligned*..... 136
 - Frame Rate* 73
 - Full Frame Digitization..... 147
 - Functions
 - Classification 121
 - Data_Present() 157
 - Get_Brightness 164
 - Get_CaptureCounter()** 163
 - Get_Contrast() 165
 - Get_Error() 124
 - Get_GPIO_Data() 169
 - Get_Hue() 166
 - Get_Sat_U() 165
 - Get_Signal_Status()** 162
 - GetVersionNumber (WIN) .. 125
 - I2C_ReadByte..... 172
 - I2C_ReadEEProm()** 174
 - I2C_Set_BR_Mode() 172
 - I2C_WriteByte() 173
 - I2C_WriteEEProm() 174
 - LumaControl() 168
 - Max_Device_Number()..... 126
 - Read_Ext_IO()..... 176
 - Read_GrabberInfo()..... 127
 - Read_Local_DWord() 177
 - Read_OrderCode()..... 128
 - Reset_CaputerCounter 163
 - Set_AGC() 137
 - Set_Brightness() 164
 - Set_BW() 135
 - Set_ChannelEx()..... 133
 - Set_CKill() 138
 - Set_Color_System()..... 129
 - Set_ColorBars() 167
 - Set_Composite() 130
 - Set_Contrast() 164
 - Set_Ext_IO() 175
 - Set_GPIO_Data()..... 169
 - Set_GPIO_Direction 169
 - Set_Hue() 166
 - Set_Image() (Win) 141
 - Set_Interlace()..... 136
 - Set_LDec() 167
 - Set_S_VideoEx() 131
 - Set_Saturation() 165
 - Start_Grabber() 155
 - Stop_Grabber() 156
 - TemporalDect()..... 139
 - Funktionen
 - Get_Video_Status()..... 130
 - Set_Image() (DOS) 187
 - Write_Local_DWord()..... 177
 - G**
 - Gr4CDLL.DLL*..... 118
 - Grabber Card
 - Installation..... 30, 56
 - H**
 - half frames..... 76
 - histogram..... 84
 - I**
 - I/O pin 25, 51, 175
 - I/O Port Testing
 - Demo Program 89
 - I²C interface 17, 43
 - I2C Interface
 - Programming..... 172
 - I²C interface 29, 55
 - Image Resolution** 16, 42
-

Information on Calling-Up the Installed Grabber	127
L	
live image	70
Live Image	81
M	
Master slots.....	30
measuring color values	85
N	
Number of Digitized Images	163
O	
Open Image on Start	81
Option-Port test (Demoprogramm)	90
P	
PCI slot	30, 56
PCI slots.....	30
PCI4GRAB.LIB	180
pciGrabber-4 Compaibility	204
PHYTEC Vision Tools Drivers and Demos	66
Pinout.....	20, 46
port pin.....	175
Ports	17, 43
Power Supply	15, 41
Power Supply from the Camera Compaibility	208
power supply output	24, 50
R	
Reducing Noise Levels	89
Relay test (Demoprogramm).....	91
Replacement fuse.....	14, 39
S	
Safety Instructions	10
Single Image	81
Snapshot	81
Snapshots	81
Storing Images Demo Program.....	92
Storing Images Under DOS....	189
Storing Parameters.....	174
S-Video camera connection	37, 63
S-Video Source	75
Synchronization	15, 41
T	
Twain driver	67
Type Casting Settings	88
V	
video power cable	37, 63
video source connections....	34, 60
video sources.....	32, 58

Document: pciGrabber-4plus/express
Document number: L-556e_6

How would you improve this manual?

Did you find any mistakes in this manual? page

Submitted by:

Customer number: _____

Name: _____

Company: _____

Address: _____

Return to:

PHYTEC Technologie Holding AG
Postfach 100403
D-55135 Mainz, Germany
Fax : +49 (6131) 9221-26

Published by

PHYTEC

© PHYTEC Messtechnik GmbH 2009

Ordering No. L-556e_6
Printed in Germany