

# User's Guide

## OPTOLCA OPTOLCA/LC



**Copyright © QUANCOM Informationssysteme GmbH**

All specifications in this manual were compiled after a thorough check, and it is not considered as a warranty for product properties. QUANCOM is not responsible for any errors or omissions contained in this user's manual, and reserves the right to make changes without notice. Passing on and duplication of this manual and the utilisation of its contents as well as the software belonging to the product are permitted only with written permission by QUANCOM.

## Table of contents

<b>1</b>	<b>Overview .....</b>	<b>6</b>
1.1	Introduction .....	6
1.2	Our experience is your profit.....	6
1.3	Customer Communication.....	6
1.4	Changes in this manual and software updates .....	6
1.5	Scope of supply .....	7
<b>2</b>	<b>Installation procedures .....</b>	<b>8</b>
2.1	System requirements .....	8
2.2	Safety precautions .....	8
2.3	Installing the QUANCOM board .....	9
<b>3</b>	<b>Technical hardware description .....</b>	<b>10</b>
3.1	General information .....	10
3.2	How the board works .....	10
3.3	Technical Data.....	10
3.4	The OPTOLCA board .....	11
3.4.1	Board outline.....	11
3.4.2	JP1: Base I/O address selection.....	12
3.4.3	JP2: Optocoupler ( common ground or common anode ) .....	12
3.4.4	JP3: (Interrupt setting) .....	12
3.4.5	JP4: (Interrupt setting) .....	12
3.4.6	Optocoupler outputs 1...16 (common anode configuration) .....	13
3.4.7	Optocoupler outputs 1...16 (common ground configuration).....	13
3.4.8	CONN1: 37 pin D-SUB Connector.....	14
3.4.9	LED (Light emitting diodes) .....	14
3.5	Hardware registers.....	15
3.5.1	INDEX Register ( write access, base address + 2 ) .....	15

---

3.5.2	Register 0_L ( write access, base address + 0) .....	16
3.5.3	Register 0_H ( write access, base address + 1).....	17
3.5.4	Register 1_H (only OPTOLCA) (write access, base address + 1) .....	17
3.5.5	Register 1_H (Read access, base address + 1).....	18
<b>4</b>	<b>How to program or use the OPTOLCA board .....</b>	<b>19</b>
4.1	Software.....	19
4.1.1	Which software to use?.....	19
4.1.2	Reading the opto-isolated digital inputs and outputs.....	20
4.1.3	Setting the optocoupler outputs .....	21
4.1.4	Read the Flip-Flops.....	22
4.1.5	Resetting the FF (Flip-Flops).....	23
4.1.6	Generating an interrupt on input change .....	23
4.1.7	Releasing the interrupt after timeout event.....	24
4.1.8	Determining the reason for the interrupt.....	24
4.1.9	Interrupt B .....	25
4.2	Input and output circuit.....	26
4.2.1	Input circuit.....	26
4.2.2	Output circuit.....	26
4.3	Diagram .....	27
<b>5</b>	<b>Softwareprogramming with the QLIB.....</b>	<b>29</b>
5.1	QLIB ( QUANCOM Driver Library ) .....	29
5.2	Software- / QLIB - Installation .....	30
5.2.1	Installing the QLIB, sources and application programs .....	30
<b>6</b>	<b>QLIB Commands.....</b>	<b>31</b>
6.1	Simple instructions (QAPI...).....	31
6.1.1	Digital write- and read functions .....	31
6.1.2	General- and special functions .....	32
6.2	Extended instructions (QAPIExt...).....	33
6.2.1	Digital write functions .....	33

6.2.2	Digital read functions .....	34
6.2.3	General functions.....	35
6.2.4	Special functions.....	36
6.3	Program examples for the QLIB.....	37
6.3.1	Controlling the Optocouplers .....	37
6.3.2	Controlling the relays .....	38
6.4	Controlling the card under Dos .....	39
6.4.1	Controlling the optocouplers .....	39
<b>7</b>	<b>Annex .....</b>	<b>40</b>
7.1	Frequently asked questions (FAQ) .....	40
7.1.1	General information .....	40
7.2	Customer Communication and Help .....	43
7.3	Technical support form .....	46
7.4	Hardware and Software configuration form.....	47
7.5	Documentation Comment Form .....	48

# 1 Overview

## 1.1 Introduction

Congratulations! You've bought a QUANCOM high quality measurement and automation board, which shows the newest update of technology and whose attributes and functions are able to compete with every other instrument and even outdoes them.

## 1.2 Our experience is your profit

We from QUANCOM are specialists for the development of hard- and software. QUANCOM has grown to become one of the leading suppliers of measuring and automation technology to industry. At our design centres QUANCOM has developed an impressive range of products.

## 1.3 Customer Communication

**QUANCOM wants to receive your comments** on our products and manuals. We are interested in the applications you develop with our products, and we want to help if you have problems with them. To make it easy for you to contact us, this manual contains comment and configuration forms for you to complete. (view "7.2 Customer Communication and Help")

## 1.4 Changes in this manual and software updates

QUANCOM - products are distinguished by their constant further development. You can see all the actual information of the changes in the README-file on the installation QUANCOM CD. You can always get more information and free software updates on our internet website.

**[WWW.QUANCOM.DE](http://WWW.QUANCOM.DE)**

## 1.5 Scope of supply

- Measuring and automation board
- User's manual
- QUANCOM CD

If a component is missing please contact your dealer. QUANCOM reserves the right to change the extent of delivery without a preliminary announcement.

## 2 Installation procedures



### 2.1 System requirements

- Personal computer: The QUANCOM boards operate in IBM-AT compatible computers with 80X86 or compatible. (for example: 80386 / 80486 / Pentium )
- Bus: Your computer must have the corresponding bus. (PCI / ISA)

### 2.2 Safety precautions

For the sake of your security and the faultless function of your new QUANCOM board mind the following advice:

- Before opening the computer please unplug it.
- Computer motherboards and components contain very delicate integrated circuit (IC) chips. To protect them against damage from static electricity, you must follow some precautions whenever you work on your computer. Use a grounded wrist strap before handling computer components. If you don't have one, touch both of your hands to a safely grounded object or to a metal object, such as the power supply case.
- Hold components by the edges and try not to touch the integrated circuit chips, leads or circuitry.
- Place components on a grounded anti-static pad or on the bag that came with the component whenever the components are separated from the system.

**! ATTENTION !** Modifications, made to the device without explicit permission from QUANCOM, lead to the loss of the operating permission and the CE certificate.

## 2.3 Installing the QUANCOM board

**CAUTION:**

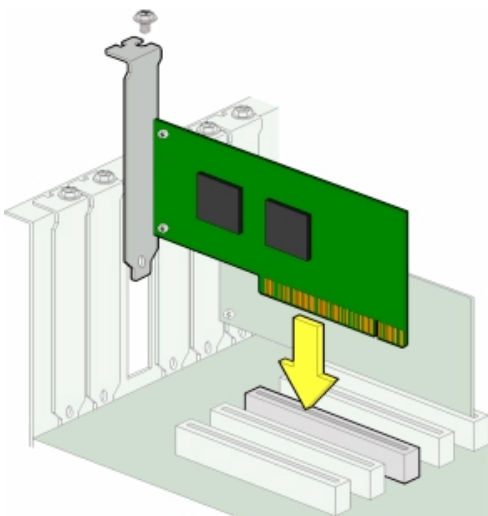
1. Always turn the system power off and remove the power cord from the wall before installing or removing any device.
2. Always pay regard to static electricity precautions.  
See „*Safety precautions*“ in chapter 2.2

1. Switch off the computer and the connected devices and unplug them.

Warning: Static electricity can destroy your computer and the board!

Discharge yourself as described in chapter 3.2 “*Safety precautions*“.

2. To open your PC you have to detach the four safety screws on the back of the case with a screw driver. Then you can pull the cover forwards. If necessary you must remove impeding cables.



3. The slots are positioned at the back side of your computer. The back wall of unused slots is covered by a small metal plate. Search for a free slot, detach its holding screw and remove the small metal plate belonging to it.

4. Position the extension card into a free slot. Pay attention that the card is set firmly in the slot.

5. Fasten the board with the screw of the small metal plate on the back wall.

6. Close the cover of your computer. Cables, that you detached during the installation, should now be reconnected.

7. Connect the cable of the board into the slot belonging to it.

## 3 Technical hardware description

### 3.1 General information

Since OPTOLCA has 16 optocoupler in and 16 optocoupler outputs, it is particularly suitable for the process control, measuring and automatic control and for all applications, for which electrically isolated inputs and outputs are needed for different potentials.

### 3.2 How the board works

The optocoupler output configuration can be easily changed, either to common mass or to common positive voltage mode. The outputs are divided in two groups where each has 8 optocouplers. The flip-flops on the optocoupler inputs can detect fast input changes and report them to your software application. An event-controlled real time processing is possible by using the interrupt channel. The interrupt driven processing of the input flip-flops is suited for very fast programs.

#### **The Interrupt channel operates as follows:**

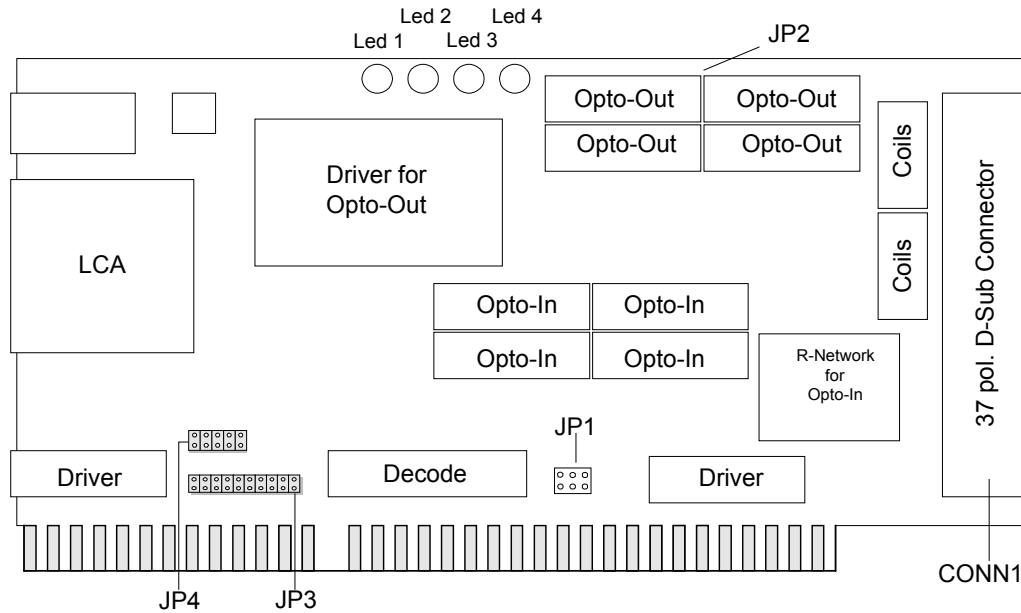
INT A is activated as soon as the status of the optocoupler inputs deviates from the previously saved a status. This function is particularly valuable for the monitoring of processes or the detection of alarm statuses.

### 3.3 Technical Data

- Number of optocoupler inputs 16
- Voltage range 5 - 30 V
- Number of optocoupler outputs 16
- Voltage range 0 - 30 V
- Current max. 50 mA per channel, ( option 100 mA )
- Connector 37-pin D-SUB
- Dimensions 148 mm x 96 mm
- Slot 16 bit ISA

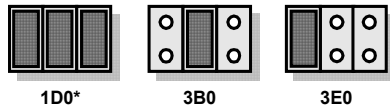
## 3.4 The OPTOLCA board

### 3.4.1 Board outline



The user may choose whether the outputs are configured as common ground or common anode. To select the type of configuration you have to move the position of the optocouplers.

### 3.4.2 JP1: Base I/O address selection



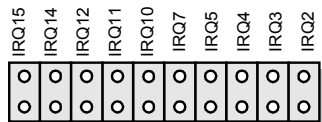
\*Default settings

### 3.4.3 JP2: Optocoupler ( common ground or common anode )

The user may choose whether the outputs are configured as common ground or common anode. To select the type of configuration you have to move the position of the optocouplers.

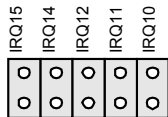
### 3.4.4 JP3: (Interrupt setting)

The JP3 allows the configuration of the source of INT A.

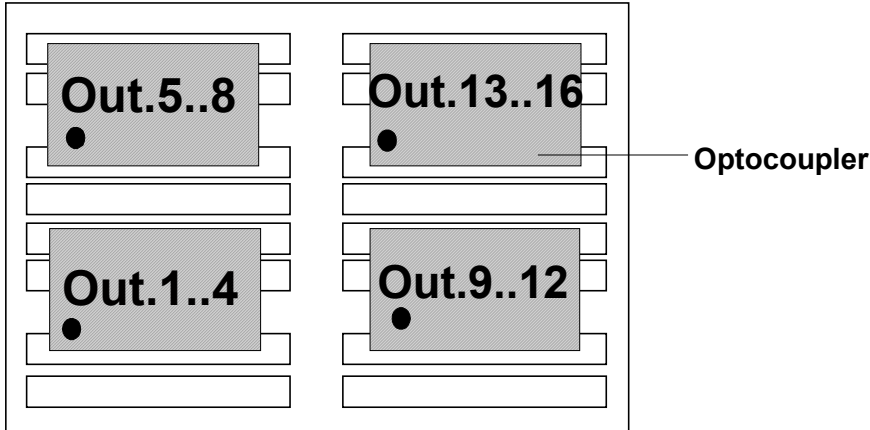


### 3.4.5 JP4: (Interrupt setting)

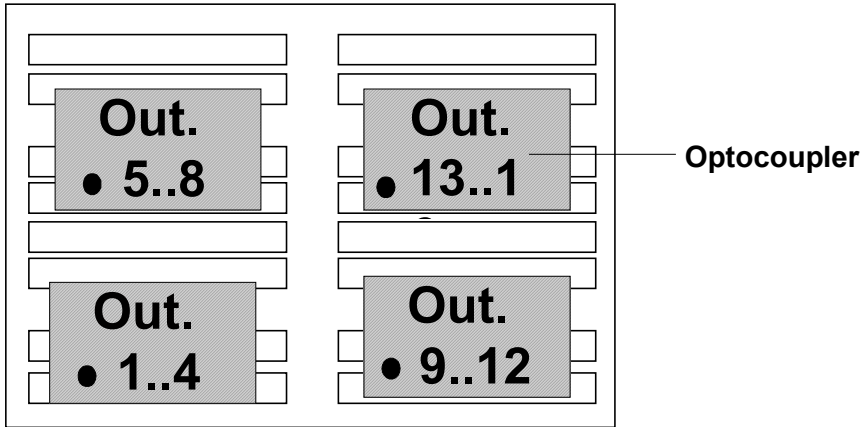
The JP4 allows the configuration of the source of INT B.



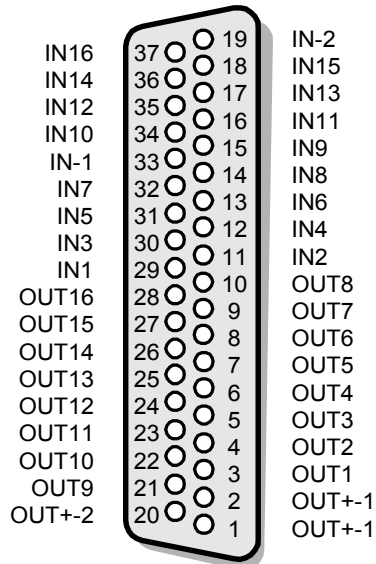
### 3.4.6 Optocoupler outputs 1..16 (common anode configuration)



### 3.4.7 Optocoupler outputs 1..16 (common ground configuration)



### 3.4.8 CONN1: 37 pin D-SUB Connector



#### Legend

OUT1...16	opto-isolated digital outputs 1...16
OUT+-1	common pin for opto-isolated digital outputs 1...8 (s. JP1)
OUT+-2	common pin for opto-isolated digital outputs 9...16 (s. JP1)
IN1...16	opto-isolated digital inputs 1...16
IN-1	common ground for opto-isolated digital input 1...8
IN-2	common ground for opto-isolated digital input 9...16

### 3.4.9 LED (Light emitting diodes)

The four LED on the OPTOLCA display the activity of some optocoupler channels.

These are:

LED1:	IN1
LED2:	IN9
LED3:	OUT1
LED4:	OUT9

### 3.5 Hardware registers

The OPTOLCA has 7 write and 5 read registers and they are divided into three register set. The table below shows an overview of all registers.

R/W ( R = Read / W = Write )	INDEX	REGISTER	Description
Before accessing the following registers, write a "0" to the index register ( base + 2 )			
<b>W</b>	0	+0	Set OUT 1..8
<b>W</b>	0	+1	Set OUT 9..16
<b>R</b>	0	+0	Read IN 1..8 ( negative logic )
<b>R</b>	0	+1	Read IN 9..16 ( negative logic )
<b>R</b>	0	+2	Read IN FF 1..8 ( negative logic )
<b>R</b>	0	+3	Read IN FF 9..16 ( negative logic )
Before accessing the following registers, write a "1" to the index register ( base + 2 )			
<b>W</b>	1	+0	Set Register 0_L
<b>W</b>	1	+1	Set Register 0_H
<b>R</b>	1	+1	Register 1_H (Timeout/INT-FF read)
Before accessing the following registers, write a "2" to the index register ( base + 2 )			
<b>W</b>	2	+1	Register 1_H (Timeout/INT-FF reset )
The following registers are not affected by the setting of the index register			
<b>W</b>	X	+2	Register INDEX set
<b>W</b>	X	+3	Clear Register IN-FF

X = don't care; FF = Flip-Flop

#### 3.5.1 INDEX Register ( write access, base address + 2 )

This register selects the register set, which is addressed by the following read or write operation. There are 3 register sets available:

<b>INDEX = 0</b>	Reading and writing of the OPTO input/outputs as well as reading of the OPTO input Flipflops
<b>INDEX = 1</b>	Interrupt- / time monitoring(Register 0_L and 0_H) and read back the timeout/INTA-Flipflops
<b>INDEX = 2</b>	Timeout- and INT A-Flipflop reset (Register 1_H)

### 3.5.2 Register 0\_L ( write access, base address + 0)

The following register allows the configuration of the interrupt, interrupt source and the input flip-flops. The following table shows the register:

Bit	Function
0	Enables Interrupt A (1=enable, 0=disable)
1	Enables Interrupt B (1=enable, 0=disable)
2	Enables the input flip-flops of the opto-isolated digital inputs; if this mode is activated the flip-flops save the occurrence of a input transition from "24V" to "0V".
3	OPTOLCA generates an Interrupt A, when the input pattern is different from the saved pattern in the flip-flop. (1=enable, 0=disable)
4...7	The Multiplexer setting for Interrupt B (e.g. OPTOI2 (Bit 7654=0001))

Some Interrupt functions are not available at on the OPTOLCA/LC board. The OPTOLCA/LC has the following register layout:

Bit	Function
0	---
1	Enables Interrupt B (1=enable, 0=disable)
2	Enables the input flip-flops of the opto-isolated digital inputs; if this mode is activated the flip-flops save the occurrence of a input transition from "24V" to "0V".
3	---
4...7	The Multiplexer setting for Interrupt B (e.g. OPTOI2 (Bit 7654=0001))

### 3.5.3 Register 0\_H ( write access, base address + 1)

This register allows the adjustment of the time control of the outputs. If within a given time there is no read or write access on the OPTOLCA card, for example, by a program error or – crash, then all outputs will be switched off. Before setting of the opto-isolated digital outputs is only possible after resetting the timeout FF (Flip-Flop) (Register 1\_H).

Bit	Function
0	Timeout value: 50 ms
1	Timeout value: 200 ms
2	Timeout value: 800 ms
3	Timeout value: 3 s
4	Timeout value: 12 s
5	Timeout value: 50 s
6	Timeout value: 210 s
7	Timeout Watchdog deactivated = 0 / activated = 1

Please add 12 ms to the configured timeout value. This prevents, a detection of timeout when the bits 0 to 6 are all set 0. With bit 7 the time control is activated. If bit 0 is set in the register 0\_L, an Interrupt will be executed every time the timeout value is exceeded (for example, the software registers a timeout event).

### 3.5.4 Register 1\_H (only OPTOLCA) (write access, base address + 1)

A write access to this register resets the timeout-counter and the interrupt FF (Flip-Flop). The value written to this register doesn't matter.

**Important: Setting the opto-outputs after a timeout event is only possible after resetting the timeout FF (Flip-Flop)!!!**

### 3.5.5 Register 1\_H (Read access, base address + 1)

This register shows information about the interrupt source.

Bit	Function
0...4	---
5	Interrupt was activated on channel INT A (input change event or timeout event)
6	---
7	Interrupt was activated by a timeout event.

This register is used within interrupt service routines (short ISR) to determine the reason for the interrupt. In operating systems which use interrupt sharing this register can be used to check whether the interrupt was requested by the OPTOLCA card. If not, the interrupt has to be passed to the next interrupt handler.

## 4 How to program or use the OPTOLCA board

### 4.1 Software

#### 4.1.1 Which software to use?

The software to be used depends on which operating system or application is being used. To access the board, following possibilities exist:

- Method 1:** Direct I/O ( access the hardware registers directly, all operating systems)
- Method 2:** High-Level programming (access the watchdog with the QLIB)

If you use **Method 1 and 2** you need the sources for the application. You are responsible for adding the programming statements in the application. For using these methods, knowledge in programming is necessary. See chapter 0, “*Technical hardware description*”, for a complete description of the hardware registers. Chapter “*QLIB: High level programming*”, shows you how to use and install the QLIB.

### 4.1.2 Reading the opto-isolated digital inputs and outputs

The relation between the optocoupler inputs and the read registers are shown below.

Read base + 0 OPTOI 1...8		Read base + 1 OPTOI 9..15	
OPTOI1	Bit 0	OPTOI9	Bit 0
OPTOI2	Bit 1	OPTOI10	Bit 1
OPTOI3	Bit 2	OPTOI11	Bit 2
OPTOI4	Bit 3	OPTOI12	Bit 3
OPTOI5	Bit 4	OPTOI13	Bit 4
OPTOI6	Bit 5	OPTOI14	Bit 5
OPTOI7	Bit 6	OPTOI15	Bit 6
OPTOI8	Bit 7	OPTOI16	Bit 7

Alternatively, it is possible to read all optocoupler inputs at once, by executing a 16 bit-read access on base address + 0. The following example shows the status the opto-isolated digital input 8:

```

unsigned int iobase = 0x1d0;    /* Base address of the card */
void read()
{
int in;
printf("OPTOI8");
outp(iobase + 2,0);           /* Register 0 (INDEX) */
in = inp(iobase + 0);        /* read 8 bit value on base address + 0 */
                             /* negative logic */

if (in & 128)
    printf(" =1 / the voltage on input is 0V ");/* isolated bit 7 (= OPTO-IN 8) */
else
    printf(" =0 / voltage on the input is greater then 0V ");
getch();
}

```

**Information:** The optical-isolated digital inputs use negative logic. If you have a zero voltage on the input the register shows high ('1'), and a voltage of 24V (voltage range depending of your input R-Network) shows a low ('0').

### 4.1.3 Setting the optocoupler outputs

The following example shows, how to set the optocoupler outputs.

```
unsigned int iobase = 0x1d0;    /* Base address of the card */
void write()
{
    outp(iobase+2,0);          /* write index register */
    outp(iobase+0,1);          /* write 8 bit value to base address +0: bit 1 on */
    getch();
}
```

#### 4.1.4 Read the Flip-Flops

The relation between the input FF (Flip-Flops) and the read registers are shown below:

Read base + 2		Read base + 3	
Flip-Flop OPTOI1	Bit 0	Flip-Flop OPTOI9	Bit 0
Flip-Flop OPTOI2	Bit 1	Flip-Flop OPTOI10	Bit 1
Flip-Flop OPTOI3	Bit 2	Flip-Flop OPTOI11	Bit 2
Flip-Flop OPTOI4	Bit 3	Flip-Flop OPTOI12	Bit 3
Flip-Flop OPTOI5	Bit 4	Flip-Flop OPTOI13	Bit 4
Flip-Flop OPTOI6	Bit 5	Flip-Flop OPTOI14	Bit 5
Flip-Flop OPTOI7	Bit 6	Flip-Flop OPTOI15	Bit 6
Flip-Flop OPTOI8	Bit 7	Flip-Flop OPTOI16	Bit 7

Alternatively it is possible to read all flip-flops with one 16-bit-read access on base address + 2.

The following example shows the status of optocoupler input 8:

```

unsigned int iobase = 0x1d0;          /* Base address of the card */

void read()
{
    unsigned int ff_value;
    outp(iobase+2,0);                /* adjust Register 0 */
    ff_value=inpw(iobase+2);         /* read FF (Flip-Flops) */
    printf("Behavior of the FF (Flip-Flops) : %04Xh\n",ff_wert);
}

```

### 4.1.5 Resetting the FF (Flip-Flops)

Writing the value 0 on base + 3 (Out base + 3,0)

Example:

```
unsigned int iobase = 0x1d0; /* Base address of the card */
void reset()
{
    outp(iobase + 3,0);      /* give Value 0 on base + 3 : */
                           /* reset all FF (Flip-Flop) */
    getch();
}
```

### 4.1.6 Generating an interrupt on input change

If want to use the detection of input changes you first have to load the comparator ( bank of FF ) with the current input value. This must be done before the interrupt A is activated.

Set bit 2 of the Register 0\_L to 1 and back to 0 to store the current inputs to the comparator.

To activate the interrupt write a value that sets bit 0 to Register 0\_L to 1.

**Example:**

With the following, current inputs are stored to the comparator and interrupt A is activated.

```
unsigned int iobase = 0x1d0; /* Base address of the card */
set_int_a_mode_a()
{
    outp(iobase+2,1);
    outp(iobase,4);
    outp(iobase,0);
    outp(iobase,1);
    getch();
}
```

### 4.1.7 Releasing the interrupt after timeout event

A further source of an interrupt is the occurrence of a timeout event. This event occurs whenever the OPTOLCA card is not accessed, via a port write or read command, within a given time. This can occur, for example due to an error in the program.

#### Example:

With the following sequence an interrupt A is generated when timing out ( T = 4 s ):

```
unsigned int iobase = 0x1d0; /* Base address of the card */
set_int_a_mode_b()
{
    outp(iobase+2,2);
    outp(iobase+1,0);
    outp(iobase+2,1);
    outp(iobase+1,142);
    outp(iobase,1);
}
```

### 4.1.8 Determining the reason for the interrupt

To determine which event has triggered the interrupt you may read the Register 1\_H.

For this purpose, the following code sequence can be used:

```
unsigned int iobase = 0x1d0; /* Base address of the card */
void InterruptServiceRoutine()
{
    outp(iobase + 2 , 1);

    if (inp(iobase + 1) & 32)
    {
        printf(„Interrupt was triggered by this card“);

        if (inp(iobase + 1) & 128)
            printf(„Reason: Timeout“);
        else
            printf(„Reason: Input has changed it’s state“);
    }
}
```

### 4.1.9 Interrupt B

One optical-isolated digital input may serve as interrupt source for interrupt B. Select a channel with writing a value to Register 0\_L. This programs the multiplexer which routes the signal from the desired input to interrupt B. The channel number is a 4 bit value written to this register. For details see the table below:

Register 0_L (s. S. 13)	Bit	7	6	5	4
	Value	128	64	32	16
OPTOIN1:		0	0	0	0
OPTOIN2:		0	0	0	1
OPTOIN3:		0	0	1	0
OPTOIN4:		0	0	1	1
OPTOIN5:		0	1	0	0
OPTOIN6:		0	1	0	1
OPTOIN7:		0	1	1	0
OPTOIN8:		0	1	1	1
OPTOIN9:		1	0	0	0
OPTOIN10:		1	0	0	1
OPTOIN11:		1	0	1	0
OPTOIN12:		1	0	1	1
OPTOIN13:		1	1	0	0
OPTOIN14:		1	1	0	1
OPTOIN15:		1	1	1	0
OPTOIN16:		1	1	1	1

**Example:**

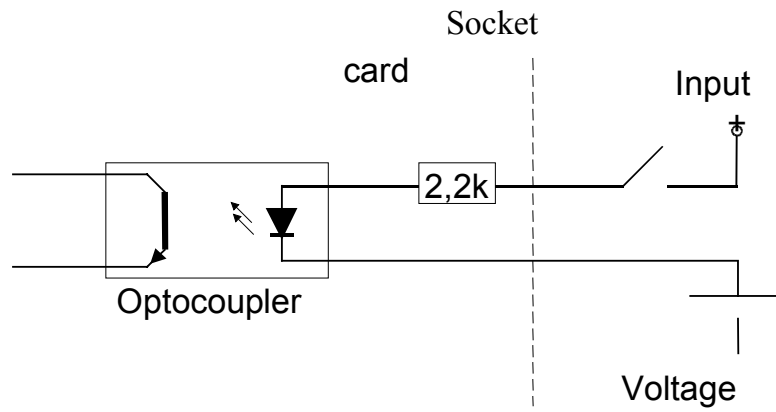
In the following example the multiplexer routes opto-channel OPTOIN3 to the interrupt B.

```

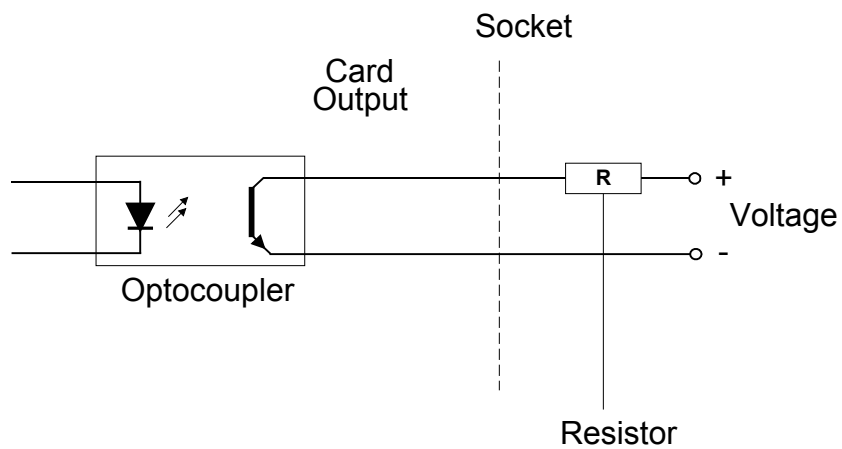
unsigned int iobase=0x1d0; /* Base address of the card */
set_int_b()
{
    outp(iobase+2,1);
    outp(iobase,2+3*16);
getch();
}
    
```

## 4.2 Input and output circuit

### 4.2.1 Input circuit



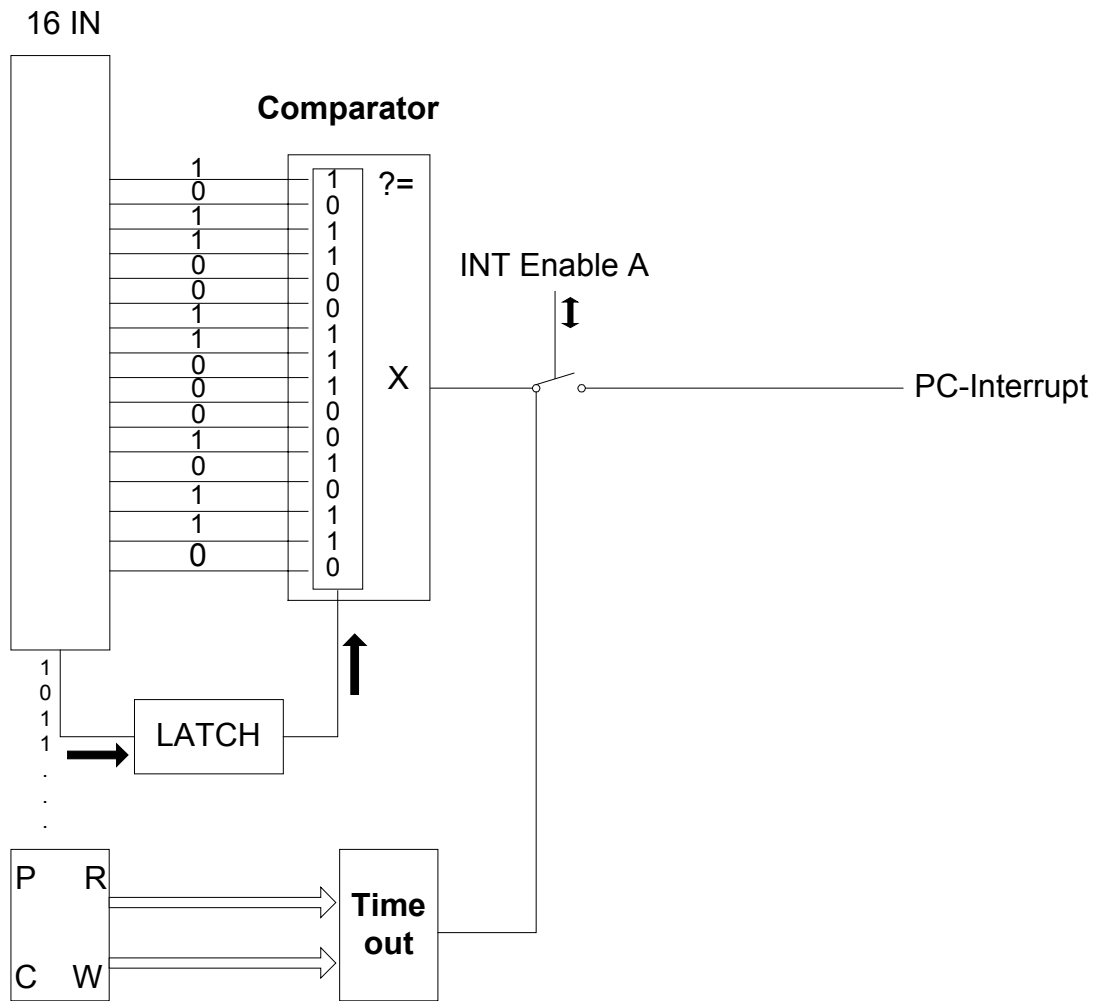
### 4.2.2 Output circuit



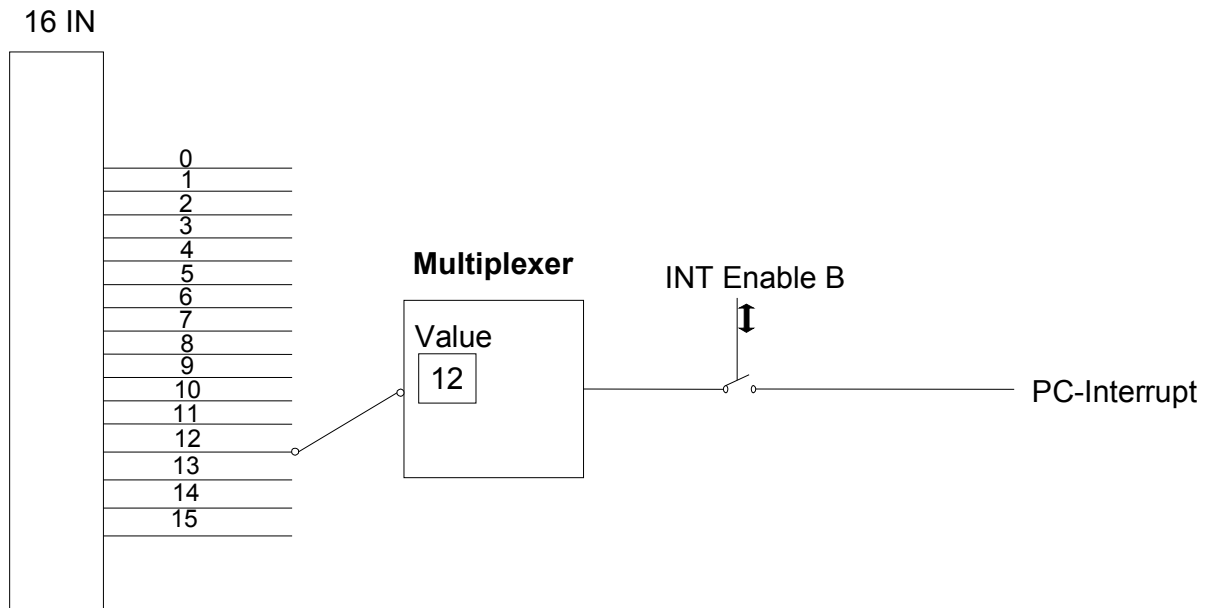
### 4.3 Diagram

After a latch command, the comparator stores the current input signal pattern in a latch. This input pattern is now continuously compared to the stored pattern. When the input lines change, an interrupt A is generated ( if this mode is activated ).To activate this mode see chapter 3.5.2 .

#### Interrupt A



## Interrupt B



If the interrupt B is activated ( see chapter 3.5.2 ) a input level change on the selected input, causes an interrupt B.

## 5 Softwareprogramming with the QLIB

### 5.1 QLIB ( QUANCOM Driver Library )

The **QLIB**, which stands for **QUANCOM Driver LIBrary**, was developed with the target to allow the simple programming of all our data acquisition products under various operating systems. So it is easy to write an application that runs under the operating systems Windows Me/98/95 and Windows XP/2000/NT4.0. This driver interface is not limited to PC boards or other I/O adapters but is also targeted towards supporting the next product generations currently being developed. The used functions and parameters are the same for all operating systems.

#### Supported operating systems:

- Microsoft Windows XP/2000/NT4.x
- Microsoft Windows ME/98/95

#### Supported compilers:

##### C / C++

- Borland C++ 3.1, 4.x, 5.x, 6.x
- Microsoft® Visual C++ 1.x, 2.x, 4.x, 5.x, 6.x

##### Pascal

- Borland Turbo Pascal

##### Delphi

- Borland Delphi

##### Basic

- Microsoft® Visual Basic 3.x, 4.x, 5.x; 6.x

#### Graphical Programming Language

- HP VEE von Hewlett-Packard
- LabView® von National Instruments

## 5.2 Software- / QLIB - Installation

### 5.2.1 Installing the QLIB, sources and application programs

1. Please insert the QUANCOM-measuring techniques-CD.  
If Windows starts the CD-Rom automatically continue with point 5.
2. Please click the **Start** button. (on the left/bottom in the Task-menu)
3. Please choose **Perform**.
4. Please perform the following command:  
**D:\Autorun**  
(If **D** is not the drive-letter of your CD-Rom drive, replace **D** with the right letter.)
5. Please choose the product name of the card and follow the installation instructions.

## 6 QLIB Commands

Make sure that the QLIB (QUANCOM Driver Library) is properly installed. For programming the UNITIMER board you need at minimum the QLIB release v1.70. For further information about the installation and how to include the necessary files in your application see the „QLIB“ documentation. This chapter describes the special commands that are required to use the UNITIMER board with the QLIB. These samples assume that the board is installed and properly set-up for use with the QLIB (QUANCOM Driver Library).

### 6.1 Simple instructions (QAPI...)

#### 6.1.1 Digital write- and read functions

##### **QAPIGetDI**

**ULONG QAPIGetDI ( ULONG cardid ULONG channel );**

With the function QAPIGetDI the condition of a 32 Bit width digital channel could be read by a DI card

##### **QAPIPutDO**

**ULONG QAPIPutDO ( ULONG cardid ULONG channel ULONG value );**

With the function QAPIExtWriteDO32 it is possible to give out a 32 Bit width digital value on a channel of the DO card

## 6.1.2 General- and special functions

### **QAPINumOfCards**

**ULONG QAPINumOfCards (void);**

With the function QAPINumOfCards it is possible to ask , which used cards are supported by the QLIB

### **QAPIGetCardInfo**

**LPCARDDATAS QAPIGetCardInfo ( ULONG cardid );**

With the function QAPIGetCardInfo it is possible to get some information about the card

### **QAPIGetCardInfoEx**

**ULONG QAPIGetCardInfoEx( ULONG cardid LPCARDDATAS lpcd );**

With the function QAPIGetCardInfoEx it is possible to get some information about the card. These will be written into the applications memory

## 6.2 Extended instructions (QAPIExt...)

### 6.2.1 Digital write functions

#### **QAPIExtWriteDO1**

```
void QAPIExtWriteDO1 ( ULONG cardhandle ULONG channel ULONG value ULONG mode );
```

With the function QAPIExtWriteDO1 it is possible to give out a 1 Bit width digital value on a channel of the DO card

#### **QAPIExtWriteDO8**

```
void QAPIExtWriteDO8 ( ULONG cardhandle ULONG channel ULONG value ULONG mode );
```

With the function QAPIExtWriteDO8 it is possible to give out a 8 Bit width digital value on a channel of the DO card

#### **QAPIExtWriteDO16**

```
void QAPIExtWriteDO16 ( ULONG cardhandle ULONG channel ULONG value ULONG mode );
```

With the function QAPIExtWriteDO16 it is possible to give out a 16 Bit width digital value on a channel of the DO card

#### **QAPIExtWriteDO32**

```
void QAPIExtWriteDO32 ( ULONG cardhandle ULONG channel ULONG value ULONG mode );
```

With the function QAPIExtWriteDO32 it is possible to give out a 32 Bit width digital value on a channel of the DO card

## 6.2.2 Digital read functions

### **QAPIExtReadDI1**

**ULONG QAPIExtReadDI1 ( ULONG cardhandle ULONG channel ULONG mode );**

With the function QAPIExtReadDI1 the condition of a 1 Bit width digital channel could be read by a DI card

### **QAPIExtReadDI8**

**ULONG QAPIExtReadDI8 ( ULONG cardhandle ULONG channel ULONG mode );**

With the function QAPIExtReadDI8 the condition of a 8 Bit width digital channel could be read by a DI card

### **QAPIExtReadDI16**

**ULONG QAPIExtReadDI16 ( ULONG cardhandle ULONG channel ULONG mode );**

With the function QAPIExtReadDI16 the condition of a 16 Bit width digital channel could be read by a DI card

### **QAPIExtReadDI32**

**ULONG QAPIExtReadDI32 ( ULONG cardhandle ULONG channel ULONG mode );**

With the function QAPIExtReadDI32 the condition of a 32 Bit width digital channel could be read by a DI card

### 6.2.3 General functions

#### **QAPIExtOpenCard**

**ULONG QAPIExtOpenCard ( ULONG cardid, ULONG devnum );**

Use the function QAPIExtOpenCard to open a board and retrieve the board handle

#### **QAPIExtCloseCard**

**void QAPIExtCloseCard( ULONG cardhandle );**

With the function QAPIExtCloseCard the board is closed

#### **QAPIExtNumOfCards**

**ULONG QAPIExtNumOfCards (void);**

With the function QAPIExtNumOfCards it is possible to ask , which used cards are supported by the QLIB

## 6.2.4 Special functions

### **QAPIExtGetCardInfo**

```
LPCARDDATAS QAPIExtGetCardInfo( ULONG cardid );
```

With the function QAPIExtGetCardInfo it is possible to get some information about the card

### **QAPIExtGetCardInfoEx**

```
ULONG QAPIExtGetCardInfoEx( ULONG cardid LPCARDDATAS lpcd );
```

With the function QAPIExtGetCardInfoEx it is possible to get some information about the card. These will be written into the applications memory

### **QAPIExtReleaseCardInfo**

```
void QAPIExtReleaseCardInfo( LPCARDDATAS carddatas );
```

With the function QAPIExtReleaseCardInfo it is possible with QAPIExtGetCardInfo to get out the asked card information

## 6.3 Program examples for the QLIB

The following program examples show you, how easy it is to develop a program to control the cards under C with the QLIB.

### 6.3.1 Controlling the Optocouplers

Program example for the optocoupler with the QLIB under C

```
#include <windows.h>
#include <stdio.h>
#include <conio.h>

#include "qlib.h"

/*=====
main program
=====*/

void main ()
{
    ULONG handle;    /* Handle of the card */

    if ((handle=QAPIExtOpenCard(OPTO16IN,0L)) == 0L)
        {
        printf("OPTOLCA couldn't be opened\n");
        return;
        }

    for (;;)
        {
        if (kbhit() != 0 && getch() == 27) break;

        printf("%04lx\n",QAPIExtReadDI16(handle,0L,0L));
        Sleep(500);
        printf("%04lx\n",QAPIExtReadDI16(handle,0L,0L));
        Sleep(500);
        printf("%04lx\n",QAPIExtReadDI16(handle,0L,0L));
        Sleep(500);
        printf("%04lx\n",QAPIExtReadDI16(handle,0L,0L));
        Sleep(500);
        }

    QAPIExtCloseCard(handle);
}
```

### 6.3.2 Controlling the relays

Program example for the relays with the QLIB under C

```
#include <windows.h>
#include <stdio.h>
#include <conio.h>

#include "qlib.h"

/*=====
main program
=====*/

void main ()
{
    ULONG handle;    /* Handle of the card */

    if ((handle =QAPIExtOpenCard(OPTOLCA,0L)) == 0L)
        {
        printf("OPTOLCA couldn't be opened\n");
        return;
        }

    for (;;)
        {
        if (kbhit()!=0 && getch()==27) break;

        QAPIExtWriteDO8(handle,0L,0x00L,0L);
        Sleep(500);
        QAPIExtWriteDO8(handle,0L,0xFFL,0L);
        Sleep(500);
        QAPIExtWriteDO8(handle,0L,0x55L,0L);
        Sleep(500);
        QAPIExtWriteDO8(handle,0L,0xAAL,0L);
        Sleep(500);
        }

    QAPIExtCloseCard(handle);
}
```

## 6.4 Controlling the card under Dos

### 6.4.1 Controlling the optocouplers

Program example for the optocoupler under C

```
#include <stdio.h>

void main()
{
    unsigned int port;

    int i;
    long int j;
    unsigned int value;

    port=0x1d0;

    if(port==0) exit(0);

    while(!kbhit()) {
        for(i=0;i<16;++i) {
            value=1<<i;

            for(j=0;j!=200000;++j);
            printf("%x %x \n",inp(port+4),inp(port+5));
        }
    }
}
```

## 7 Annex



### 7.1 Frequently asked questions (FAQ)

#### 7.1.1 General information

**Can I have any problems with network boards, sound cards, system components or other expansion boards?**

Yes, according to how you have set the I/O address of your QUANCOM and the other component. You may encounter a resource conflict if the QUANCOM board and another component are using the same I/O address. Either change the I/O address of the QUANCOM board or that of the other component.

Problems with ISA boards running under Windows 98/95 and Windows XP/2000/NT

**Why is the “Control Panel” board configuration dialog “QLIB” empty?**

- There are no drivers installed for a QUANCOM ISA board.

**After installation I get the message “QLIBNDRV.SYS not found“ or “QLIBNDRV.VXD not found“ . What can I do?**

- Check that the QLIB is installed properly. For further information about the installation process and the general programming with the QLIB please see the „QLIB“ manual which is included on the installation CD.
- If you use a QUANCOM ISA board, check if the drivers for the QUANCOM board are installed. and if the driver is configured properly. (**“Control panel” => System**)

**After the software installation I get the message „Direct-IO interface cannot be initialised” or “qmulti32.dll could not be initialised“. What can I do?**

- Check that the QLIB is installed properly. For further information about the installation process and the general programming with the QLIB please see the „QLIB“ manual which is included on the installation CD.
- If you use a QUANCOM ISA board check if the drivers for the QUANCOM board are installed. and if the driver is configured properly. (**“Control panel” => System**)

**Why does QAPIExtOpenCard ( ... ) return the value 0, although the card is installed?**

- Check that the QLIB is installed properly. For further information about the installation process and the general programming with the QLIB please see the „QLIB“ manual which is included on the installation CD.
- The card is not configured. (**“Control panel” => QLIB**, see the „QLIB“ manual which is included on the installation CD)

**Why do I get the message "Driver QLIBNDRV.SYS" or “Driver QLIBNDRV.VXD” could not be loaded?**

- Check that the QLIB is installed properly. For further information about the installation process and the general programming with the QLIB please see the „QLIB“ manual which is included on the installation CD.
- The driver for the QUANCOM board was not loaded. (**Control Panel => System** )

**Windows XP/2000/NT: Must the QLIB be installed with Administrator-right?**

- Yes, always install the QLIB with administrator rights.

---

**Windows XP/2000/NT: Why do I get the message “Driver could not be installed” during the installation?**

- Installation was made without administrator rights.

**Windows XP/2000/NT: Why do I get the message "Driver QLIBNDRV.SYS could not be loaded"?**

- Installation of the drivers has failed, because the QLIB was not installed with administrator rights.
- QLIB-Software was installed on a network drive. Always install the QLIB on your local drive.

**Windows XP/2000/NT: How can I install the driver QLIBNDRV.SYS manually?**

If the QLIBNDRV.SYS failed to install, it may be necessary to install the driver manually.

- Search on the installation CD in the directory “Tools” for the tool “instdrv.exe”. With this tool you can install and reinstall the driver manually.
- Please call this tool with the following command line parameters:

```
instdrv qlibndrv d:\directory\qlibndrv.sys .
```

Replace **d:\directory\** with the drive and the file, where the driver “qlibndrv.sys” is located.

- Go to “**Start -> Settings ->Control panel ->( Administrative Tools / Windows 2000 only ) -> Drivers**” change the start type to “automatic”, then click on the **start** button. For the changes to become active please restart the system.

**Why must I restart the driver after every reboot of the computer?**

The starting type of the driver is set to “Manual”. If you want you can change this setting on „Automatic“ to start the driver on every reboot of the system.

## 7.2 Customer Communication and Help



### You need help?

If you don't know how to go on during the installation or operation of your QUANCOM board please first read this user's guide.

### ! Tip !

On the QUANCOM installation CD you can find a ASCII – Text – file README.TXT, which includes changes made after printing this user's manual.

### ! IMPORTANT !

If you have further questions please contact our support team and have the following information handy:

- Exact type of board
- Operating system, hardware equipment and Bus - System
- Name and Version of the program, where the error is reported.
- A detailed failure description. To make sure, please try to reproduce the failure, describe it as exact as possible, and which steps led to this failure.

**Who can you contact?**

The QUANCOM internet website

[WWW.QUANCOM.DE](http://WWW.QUANCOM.DE)

Per Fax

**+49 22 36 / 89 92 - 49**

Per e-mail:

**support@quancom.de**

Address:

**QUANCOM INFORMATIONSSYSTEME**

**GmbH**

**In der Flecht 14**

**D-50389 Wesseling**

**Germany**

If you need urgent help call:

QUANCOM Hotline Germany

**+49 22 36 / 89 92 - 20**

Monday-Thursday

from 9:00 to 18:00

Friday

from 9:00 to 17:00

**Actual drivers**

You can find the newest Version of QUANCOM software on our internet website <http://www.quancom.de>. You can also find a lot of information and „Frequently asked questions (FAQ's)". Before you contact the QUANCOM support, please check if you are using the newest software version of the QUANCOM software.

## Repair

If you are not sure whether your QUANCOM board is defective please call the QUANCOM Hotline:

Tel.: **+49 22 36 / 89 92 – 20**

Before you send us the QUANCOM board to be repaired call:

Tel.: **+49 22 36 / 89 92 – 20**

If you send us your QUANCOM board, please use original package or any other suitable package, to protect the contents against transport damage. You also need to send us a copy of the original bill and the RMA number.

You can shorten the repair time by sending us an exact failure description, so that a faster failure search is possible. Send your QUANCOM board directly to the service department of QUANCOM Informations-systeme GmbH.

### 7.3 Technical support form

If you have internet access please enter the following URL in your browser: <http://www.quancom.de/quancom/qshop.nsf/techniksupport?OpenForm&eng> else photocopy this form and use the copy of this form as a reference for your current configuration. Complete this form before contacting QUANCOM Informationssysteme GmbH for technical support and our applications engineers may answer your questions more efficiently. If you are using any other QUANCOM hardware or software products please add them to this configuration form. Include additional pages if necessary.

Name: \_\_\_\_\_

Company: \_\_\_\_\_

Address: \_\_\_\_\_

Phone: \_\_\_\_\_

Fax: \_\_\_\_\_

Computer brand / Processor: \_\_\_\_\_

Operating system: \_\_\_\_\_

Display adapter: \_\_\_\_\_

Mouse: \_\_\_\_\_

QUANCOM board \_\_\_\_\_

Other adapters installed: \_\_\_\_\_

Hard disk (capacity, size MB, free): \_\_\_\_\_

The problem is: \_\_\_\_\_

List any error messages: \_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

The following steps cause the problem to recur:

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

## 7.4 Hardware and Software configuration form

This form helps you to list your hardware and software settings. Complete this form each time you change your software or hardware configuration, and use this form as a reference for your current configuration. Complete this form accurately before contacting QUANCOM Informationssysteme GmbH for technical support, so that our applications engineers can answer your questions more efficiently.

• **QUANCOM Product:**

Name / Name of board \_\_\_\_\_  
Interrupt Level \_\_\_\_\_  
DMA Channel \_\_\_\_\_  
Base I/O Address \_\_\_\_\_  
Operating system \_\_\_\_\_

• **Other information**

Computer brand and Model \_\_\_\_\_  
Processor \_\_\_\_\_  
Clock Frequency \_\_\_\_\_  
Type of Video Board Installed \_\_\_\_\_  
DOS Version \_\_\_\_\_  
Programming Language \_\_\_\_\_  
Programming Language Version \_\_\_\_\_

• **Other Boards in System**

Base I/O Address of other Boards \_\_\_\_\_  
DMA Channels of other Boards \_\_\_\_\_  
Interrupt Level of other Boards \_\_\_\_\_

## 7.5 Documentation Comment Form

QUANCOM Informationssysteme GmbH would like you to comment on the documentation supplied with our products. This information helps us to provide you with quality products to meet your needs. Include additional pages if necessary.

Title: **OPTOLCA**

Edition Date: **10.08.04 16:16**

Please comment on the completeness, clarity, and organisation of the manual. If you find errors in the manual, please record the page numbers and describe the errors. Thank you for your help.

Name: \_\_\_\_\_

Company: \_\_\_\_\_

Address: \_\_\_\_\_

Phone: \_\_\_\_\_

Fax: \_\_\_\_\_

Comment: \_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

Mail to: [support@quancom.de](mailto:support@quancom.de)

Fax to: +49 2236 89 92 49

Address: **QUANCOM Informationssysteme GmbH  
In der Flecht 14,  
50389 Wesseling  
Germany**

**registered trade-mark:**

Windows/386 is trade-mark of Microsoft  
XT and PS/2 are trade-marks and IBM, OS/2 and AT are registered of the International Business  
Machines Corporation.

National Instruments, LabVIEW are registered trade-mark of National Instruments Corporation  
MS, MS-DOS, Microsoft, Visual Basic, Windows, Windows NT are registered trade-mark  
of Microsoft Corporation

By other product- and company names, that are mentioned in this manual, it may deal with trademarks  
of the respective owners.