CPCI-DIO1616

CompactPCI-Digital I/O-Board

Software Manual

to Product I.2309.xx

esd electronic system design gmbh
Vahrenwalder Str. 207 • 30165 Hannover • Germany
www.esd-electronics.com • Fax: 0511/37 29 8-68
Phone: 0511/37 29 80 • International: +49-5 11-37 29 80

NOTE

The information in this document has been carefully checked and is believed to be entirely reliable. **esd** makes no warranty of any kind with regard to the material in this document, and assumes no responsibility for any errors that may appear in this document. **esd** reserves the right to make changes without notice to this, or any of its products, to improve reliability, performance or design.

esd assumes no responsibility for the use of any circuitry other than circuitry which is part of a product of **esd** gmbh.

esd does not convey to the purchaser of the product described herein any license under the patent rights of **esd** gmbh nor the rights of others.

esd electronic system design gmbh
Vahrenwalder Str. 207
30165 Hannover
Germany

Phone: +49-511-372 98-0
Fax: +49-511-372 98-68
E-mail: info@esd-electronics.com
Internet: www.esd-electronics.com

USA / Canada:
esd electronics Inc.
525 Bernardston Road
Suite 1
Greenfield, MA 01301
USA

Phone: +1-800-732-8006
Fax: +1-800-732-8093
E-mail: us-sales@esd-electronics.com
Internet: www.esd-electronics.us

Manual file:	I:\Texte\Doku\MANUALS\CPCI\DIO1616\CPCI-DIO1616_12S.en9
Date of print:	2008-04-24

Described software:	**Described software revision:**
Driver for QNX6 (esd order no.: I.2309.32)	Rev.: 1.0.5
Driver for: Windows 2000 Windows XP Windows Vista (32-bit) (esd order no.: I.2309.12)	At present only a beta version (0.1.0) is available, only the following commands are supported: - `DIOCTL_SEND` - `DIOCTL_RECEIVE` - `DIOCTL_RECEIVE_OUTPUT`

Changes in the software and/or the manual

Chapter	Changes in this manual from the previous version
2.1	Description of the installation of the device driver under QNX6. Change of the entry
2.2	Description of the installation of the device driver under Windows inserted
4.1	`typdef uint64_t DIOTICK` inserted
4.3	`DIOCTL_RECEIVE_OUTPUT` und `DIOCTL_RECIRQ_FLUSH` in the overview of commands
4.9	Description of command `DIOCTL_RECEIVE_OUTPUT`
4.10	Description of the command `DIOCTL_RECIRQ_FLUSH`

This page has intentionally been left blank.

Contents

Page

1. Introduction 7

2. Starting the Driver 7
2.1 Installation of the Device Driver Under QNX6 7
2.2 Installation of the Device Driver Under Windows 8

3. Using the Driver 11

4. Functions 12
4.1 Structures 12
4.2 Assignment of Inputs and Outputs to Parameter Bits 12
4.3 Overview of Commands 13
4.4 DIOCTL_SEND - Setting the Digital Outputs and the User LED 14
4.5 DIOCTL_RECEIVE - Receiving the Digital Inputs 15
4.6 DIOCTL_RECIRQ - Receiving Interrupt Source 16
4.7 DIOCTL_IRQSET - Initializing the Interrupts 17
4.8 DIOCTL_TIMERES - Determining the Time Basis 18
4.9 DIOCTL_RECEIVE_OUTPUT - Reading the Digital Outputs 19
4.10 DIOCTL_RECIRQ_FLUSH - Flush the Input Event Queue 20

This page has intentionally been left blank.

1. Introduction

This manual describes the driver software of the CompactPCI-module CPCI-DIO1616 for the operating systems QNX6 and Windows 2000/XP/Vista(32-bit).

2. Starting the Driver

2.1 Installation of the Device Driver Under QNX6

In order to start the driver you have to enter

```
devio-pcidio &
```

Note: The driver can only be started with root rights!

The driver should be started via one of the start scripts `/etc/system` or `/etc/rc.d` or with the PCI-enumerator.

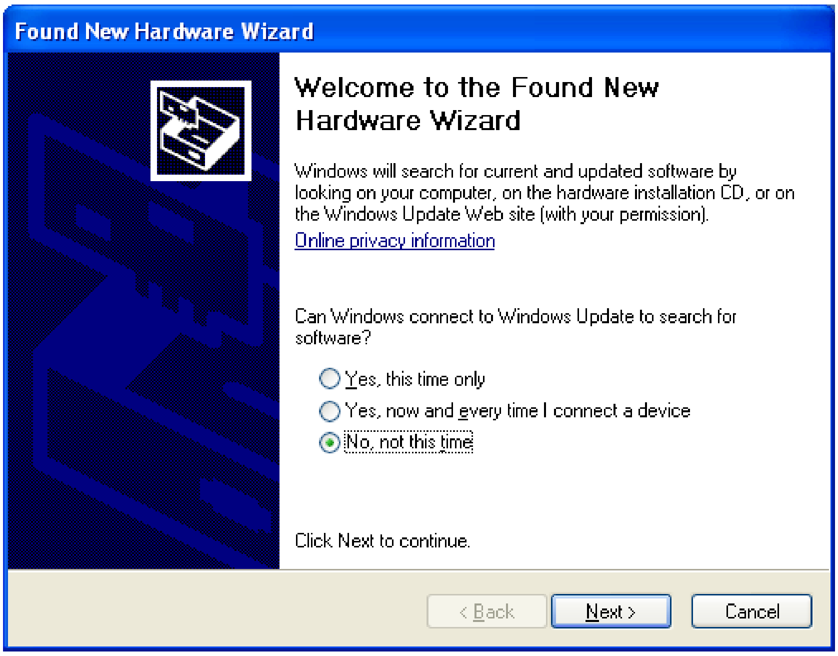
2.2 Installation of the Device Driver Under Windows

Note: In order to install the device driver for Windows 2000/XP/Vista (32-bit) the user must have administrator privileges.

The *Hardware Wizard* of the Windows operating system will guide you through the installation of the drivers. The installation will be described taking Windows XP as an example. The *Hardware Wizard* windows may vary for other Windows systems.

The hardware installation of the module must be done before Windows XP is started.
Windows will indicate the detection of the new hardware by starting the *Found New Hardware Wizard*. Please follow the instructions found there.

Select *No, not this time* from the available options and proceed by clicking the *Next* button.



Found New Hardware Wizard

Welcome to the Found New Hardware Wizard

Windows will search for current and updated software by looking on your computer, on the hardware installation CD, or on the Windows Update Web site (with your permission).

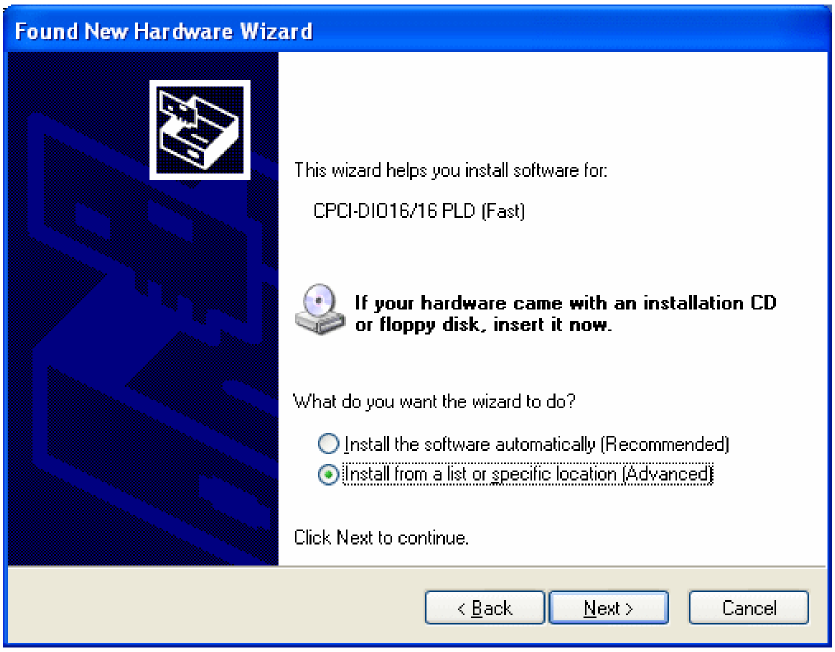
Online privacy information

Can Windows connect to Windows Update to search for software?

○ Yes, this time only
○ Yes, now and every time I connect a device
● No, not this time

Click Next to continue.

In the following dialogue box choose *Install from a list or specific location (Advanced)* before you click the *Next* button.



This wizard helps you install software for:

CPCI-DIO16/16 PLD (Fast)

If your hardware came with an installation CD or floppy disk, insert it now.

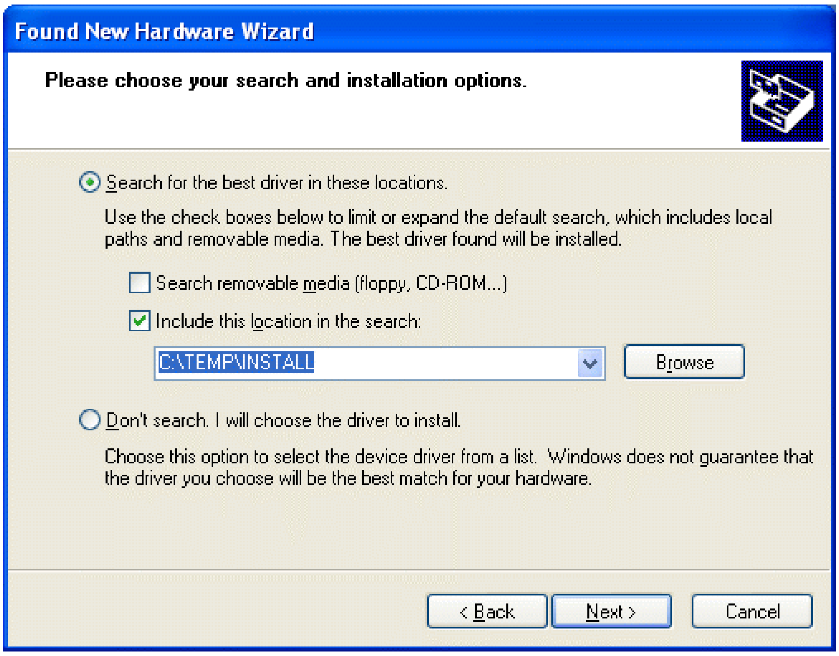
What do you want the wizard to do?

○ Install the software automatically (Recommended)

● Install from a list or specific location (Advanced)

Click Next to continue.

Select *Search for the best driver in these locations* and choose *Search removable media* if you install from a CD or choose *Include this location in the search* and browse for the location of the driver files. By clicking the *Next* button the Wizard starts installing the device driver.



Please choose your search and installation options.

● Search for the best driver in these locations.

Use the check boxes below to limit or expand the default search, which includes local paths and removable media. The best driver found will be installed.

□ Search removable media (floppy, CD-ROM...)

☑ Include this location in the search:

C:\TEMP\INSTALL

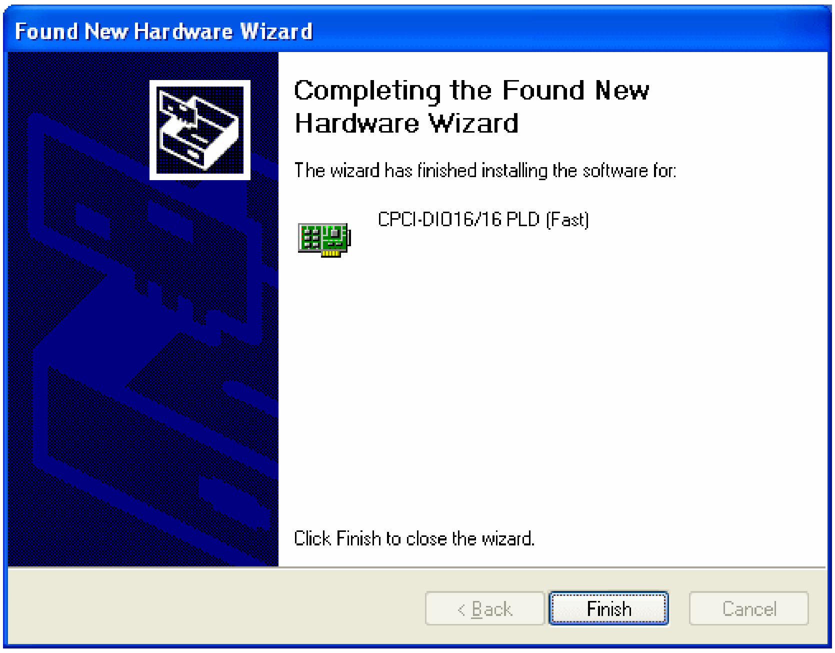
○ Don't search. I will choose the driver to install.

Choose this option to select the device driver from a list. Windows does not guarantee that the driver you choose will be the best match for your hardware.

The driver package is not signed by WHQL (Windows Hardware Quality Labs). Depending on the configuration of the computer a warning concerning the driver might appear during the installation of a driver (non-WHQL certified) .
Click on the *Yes* button and resume the driver installation.

Now, the device driver software is installed.

After copying the necessary driver files to the system, Windows displays a message indicating that the installation has been successful. Click *Finish* to complete the installation.



Found New Hardware Wizard

Completing the Found New Hardware Wizard

The wizard has finished installing the software for:

CPCI-DIO16/16 PLD (Fast)

Click Finish to close the wizard.

After the driver installation has been completed, you may open the device manager to verify the correct installation and configure driver parameters.

3. Using the Driver

The drivers are started with the following calls:

QNX6: `devctl()`

Windows XP: `DeviceIoControl()`

The commands and parameters are described under header `esdio.h`.

The example program `diotest.c` is included in the delivery as source code.

4. Functions

4.1 Structures

The commands described below use the following structures for their parameters:

```
typedef struct
{
   uint32_t mask;
   uint32_t data;
   uint64_t time;
} DIODATA;

typedef struct
{
   uint32_t edgeRising;
   uint32_t edgeFalling;
} DIOIRQ;

typdef uint64_t DIOTICK;
```

4.2 Assignment of Inputs and Outputs to Parameter Bits

Bit	Assignment	Level Assignment
31 : 17	not evaluated : not evaluated	-
16	*User-LED-Bit*	‘0’ -> LED off ‘1’ -> LED on
15 : 0	*OUT15* : *OUT0*	‘0’ -> output off ‘1’ -> output on

Table 4.2.1: Assignment of output registers

Bit	Assignment	Level Assignment
31 : 20	insignificant : insignificant	-
19 18 17 16	*Error_12-15* *Error_8-11* *Error_4-7* *Error_0-3*	‘0’ -> no output error ‘1’ -> output error
15 : 0	*IN15* : *IN0*	‘0’ -> input inactive ‘1’ -> input active

Table 4.2.2: Assignment of input registers

4.3 Overview of Commands

Command	Function
`DIOCTL_SEND`	Setting the digital outputs and the user LED
`DIOCTL_RECEIVE`	Reading the digital inputs and the error messages of the outputs
`DIOCTL_RECIRQ`	Reading the interrupt source
`DIOCTL_IRQSET`	Initializing the interrupts
`DIOCTL_TIMERES`	Determining the time basis of the returned timestamp
`DIOCTL_RECEIVE_OUTPUT`	Get the state of the digital output pins
`DIOCTL_RECIRQ_FLUSH`	Flush the digital input event queue

Table 4.3.1: Overview of the implemented commands

Note: Only the following commands are currently supported by the Windows driver (Windows 2000/XP/Vista):

```
DIOCTL_SEND
DIOCTL_RECEIVE
DIOCTL_RECEIVE_OUTPUT
```

4.4 DIOCTL_SEND - Setting the Digital Outputs and the User LED

DIOCTL_SEND

Name: **DIOCTL_SEND** - Setting the digital outputs and the user LED

Input parameters: *DIODATA.mask*: bit(x)=1 => output x to send
- masks the output bits which are to be set

DIODATA.data: output state for bits matched in mask
- sets the outputs to ‘0’ or ‘1’

Output parameters: none

Description: This command has been designed to set the outputs. Outputs which are to be changed have to be masked first.

After reset or system boot all outputs are switched off.

4.5 DIOCTL_RECEIVE - Receiving the Digital Inputs

DIOCTL_RECEIVE

Name: **DIOCTL_RECEIVE** - Receiving the digital inputs and error messages

Input parameters: *DIODATA.mask*: bit(x)=1 => input x to receive
- masks the input bits which are to be received

Output parameters: *DIODATA.mask*: echo of input mask
- echoed value of input mask

DIODATA.data: input state for bits matched in mask; unmatched bits set to zero
- status of received input bits or error messages of the output drivers

Description: This command has been designed to receive digital inputs and evaluate the error signals of the output drivers. Via a mask the bits are selected whose levels are to be received. The value of the mask is echoed.
The current status of the input bits can be read under *DIODATA.data*. The status of bits which have not been masked is always set to '0'.

Status of bits to be evaluated:

input active (power supplied):	bit = '1'
error signal active	bit = '1'
input inactive (no or too low input voltage):	bit = '0'
error signal inactive	bit = '0'

4.6 DIOCTL_RECIRQ - Receiving Interrupt Source

DIOCTL_RECIRQ

Name: **DIOCTL_RECIRQ** - Receiving interrupt source

Input parameters: none

Output parameters:

DIODATA.mask: bit(x) = 1 => edge for input x detected
- masks input and error bits for interrupt evaluation

DIODATA.data: input state for bits matched in mask; unmatched bits set to zero
- input or error bit which triggered interrupt

DIODATA.time: timestamp of detected edge(s)
- the time basis can be determined via DIOCTL_TIMERES

Description: Via this command the input bit(s) or error bits which triggered an interrupt are determined.
The timestamp of the interrupt can be read from *DIODATA.time*.

The call is blocked until the desired edge(s) appear(s).

4.7 DIOCTL_IRQSET - Initializing the Interrupts

DIOCTL_IRQSET

Name: **DIOCTL_IRQSET** - Initializing the interrupts

Input parameters:

DIOIRQ.edgeRising:	bit(x) = 1 =>	interrupt enable for rising edge on input x
	bit(x) = 0 =>	interrupt disable for rising edge on input x
DIOIRQ.edgeFalling:	bit(x) = 1 =>	interrupt enable for falling edge on input x
	bit(x) = 0 =>	interrupt disable for falling edge on input x

Output parameters: none

Description: Via this command the interrupts for falling and/or rising edge on input x are enabled or disabled. After starting up all interrupts are disabled.

4.8 DIOCTL_TIMERES - Determining the Time Basis

DIOCTL_TIMERES

Name: **DIOCTL_TIMERES** - Determining the time basis of the timestamp

Input parameters: none

Output parameters: *DIOTICK resolution*: clock cycles per second
- number of clock cycles per second

Description: Via this command the time basis of the clock cycle of the controller clock can be determined for reasonable timestamp evaluation.

4.9 DIOCTL_RECEIVE_OUTPUT - Reading the Digital Outputs

DIOCTL_RECEIVE_OUTPUT

Name: **DIOCTL_RECEIVE_OUTPUT** - Get the state of the digital output pins

Input parameters: *DIODATA.mask*: bit(x)=1=> output pin x to receive

- masks the output bits which are to be read

Output parameters: *DIODATA.mask*: echo of input mask

- echoed value of input mask

DIODATA.data: output state for pins matched in mask (unmatched bits are set to zero)

- status of received output bits, matched in mask, unmatched bits are set to ‘0’.

Description: This command has been designed to read the digital outputs. A mask defines the bits whose levels are to be received.
The current status of the output bits can be read in *DIODATA.data*.
The status of the unmatched bits is always set to ‘0’.

4.10 DIOCTL_RECIRQ_FLUSH - Flush the Input Event Queue

DIOCTL_RECIRQ_FLUSH

Name: **DIOCTL_RECIRQ_FLUSH** - Delete the input event queue

Input parameter: none

Output parameter: none

Description: This command deletes the interrupt events of the current handle. An opened handle can buffer up to 256 events.