Technical Specification

PXI / PCI 3051

CAN Interfaces

User Manual

Version 2.0

GOPEL electronic

Get the total Coverage!

GOEPEL electronic GmbH
Goeschwitzer Str. 58/60
D-07745 Jena
Phone: +49-3641-6896-597
Fax: +49-3641-6896-944
E-Mail: ats_support@goepel.com
http://www.goepel.com

printed: 27.04.2011

Issue: April 2011

1 Installation of the Board

1.1 Hardware Installation

Please make absolutely certain that all of the installation procedures described below are carried out with your system switched off.

The PCI™-, CompactPCI™- or PXI™ system is to be opened according to its conditions. A free slot is to be selected in your system.
Now, the slot cover is to be taken away from the slot selected. To do this, unscrew the two fixation screws and remove the cover from the slot.

(If it is necessary to exchange transceiver modules, pay attention to the general rules to avoid electrostatic charging.
Transceiver modules must never be removed or mounted with the power switched on! Additionally, the right alignment is absolutely required.)

When installing the board, touch it at its edges only.
Never touch the surface of the board, because otherwise it may be destroyed by electrostatic charges.

Insert the board carefully into the prepared slot. Use the lever at the front plate in order to push in the board finally.

When the board has been inserted properly, it is to be fixed by means of the two screws at the front plate. Now, the board has been installed correctly.
Afterwards, carry out the operations required at the system to make it ready for operation anew.

1.2 Driver Installation

1.2.1 Windows Device Driver

Due to the plug and play capability of Windows® 2000/ XP, for every newly recognized hardware component a driver installation is started automatically via the hardware assistant. The hardware assistant can carry out the installation of the device driver by using the *inf* file contained on the enclosed CD.

It is not absolutely essential to restart the system.

At present, the available device driver only supports Windows® 2000/ XP systems.

If you want to create your own software for the boards, you possibly need additional files for user specific programming (**.LLB*, **.H*). These files are not automatically copied to the computer and have to be transferred individually from the supplied CD to your development directory.

The I/O base address is generated during the boot operation of the system and is written into the configuration area of the board.
A manual setting is not necessary.

Interrupts and DMA channels are not required for these boards.

1.2.2 VISA Device Driver

First step

Copy the *VISA_Driver* directory of the delivered CD to your hard disk. (Recommendation: Complete directory to *C:* \)

Second step

WindowsNT :

Then, open the *C:* *VISA_Driver* *PXI3051* *Installation* subdirectory and install the *PXI3051_NT4.inf* file (mark this file and open a popup menu by the right mouse button; in the popup menu, execute the *Installation* menu entry).

During installing the **.inf* file, no information regarding installation is given.

Windows98, Windows2000, WindowsXP :

Due to the plug and play capability, for every newly recognized hardware component a driver installation is started automatically via the hardware assistant. Follow the instructions. Enter as target directory the one which contains the **.inf* file (according to recommendation: *C:* *VISA_Driver* *PXI3051* *Installation*).

For Windows98, select the *PXI3051_9x.inf* file, and for Windows2000/XP the *PXI3051_NT5.inf* file.

LabViewRT :

For operating PXI/ PCI 3051 boards under the RT operating system, use the *P3051_RT.inf* file in the *C:* *VISA_Driver* *PXI3051* *Installation* directory.

Copy this file to the *ni-rt* *system* folder of the embedded controller (recommendation: copy by the NI Measurement Explorer).

If you intend to create a *startup.rtexe* later, copy also the *cvi_lvrt.dll* file tot he *ni-rt* *system* folder.

Third step:

Reboot your computer to complete installation.

After driver installation, you can check whether the boards are properly imbedded by the system:

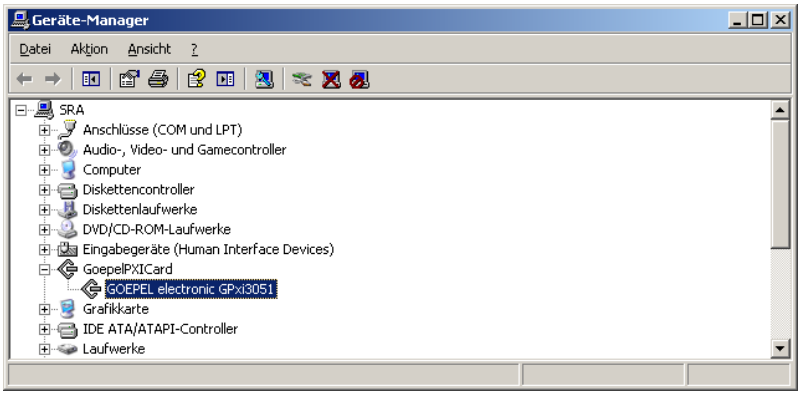

Figure 1-1: Windows

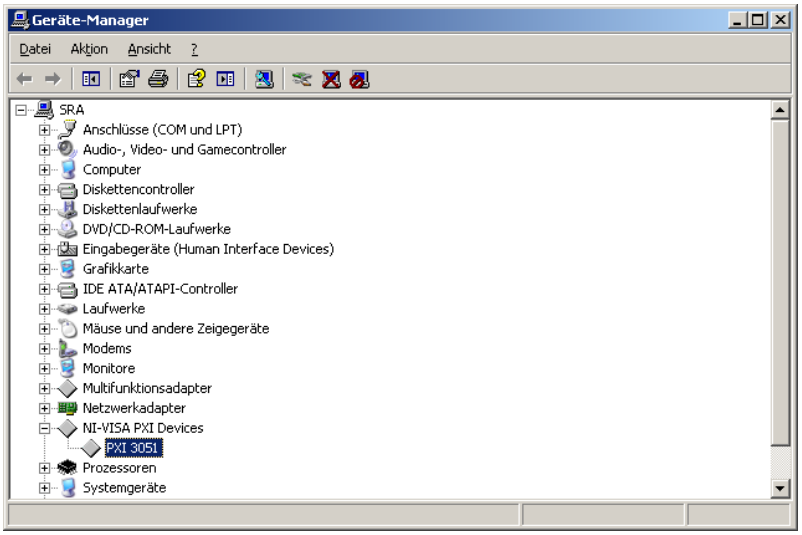

Figure 1-2: Visa

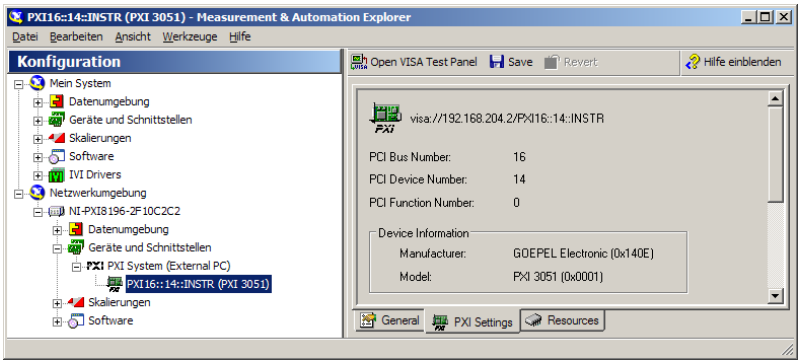

Figure 1-3: Visa RT

2 PXI / PCI 3051 Hardware

2.1 Definition

The PXI 3051/ PCI 3051 CAN interface boards are communication boards of GOEPEL electronic GmbH.

These boards with up to four CAN interfaces are used in general control technology, especially for applications in automotive technology.

In this User Manual, Controller means ALWAYS the microcontroller assigned to a CAN interface
(with the exception of the “CAN Controller” designation on the front panel for the entire board).

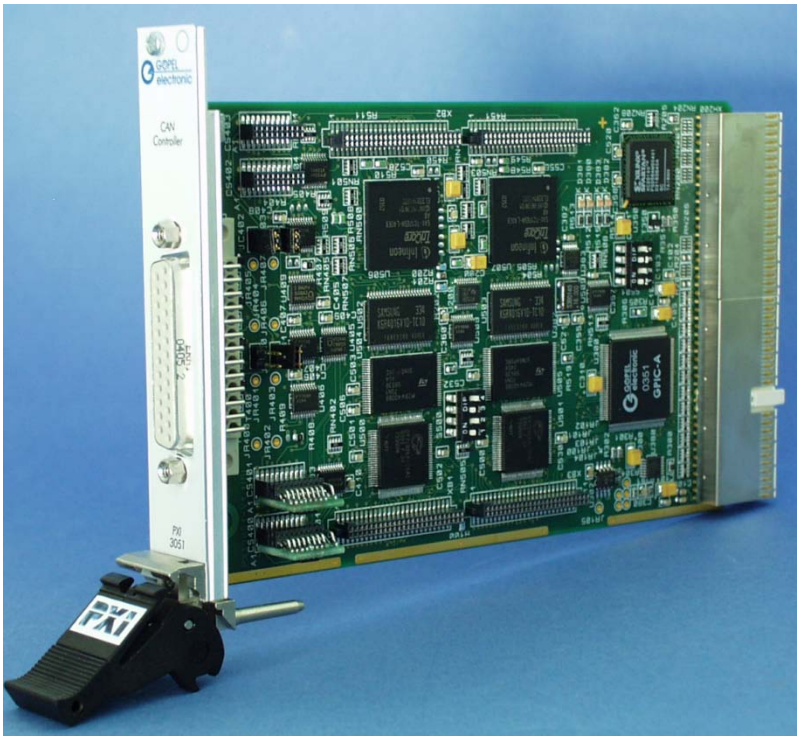

Figure 2-1: PXI 3051 Basic board

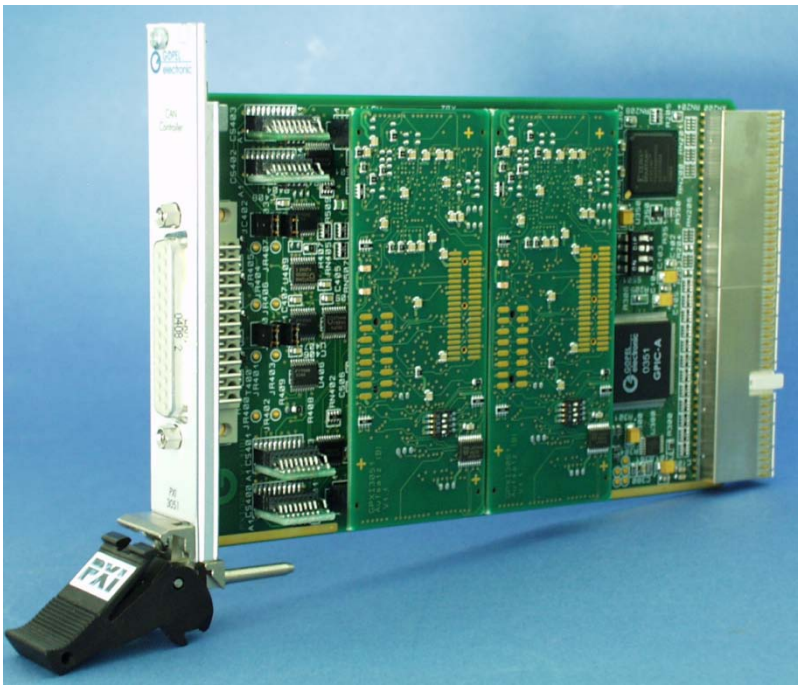

Figure 2-2:
PXI 3051 with 4 x CAN

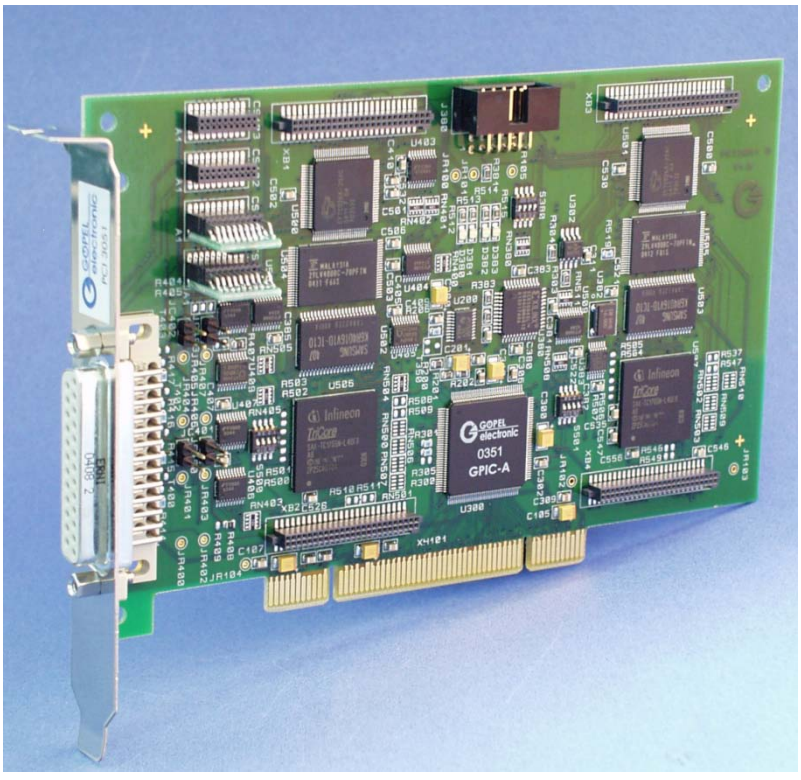

Figure 2-3:
PCI 3051 Basic board

2.2 Technical Data

2.2.1 General

The PXI 3051 communication board is a plug-in board developed for the PXI™ bus (**P**CI e**X**tensions for **I**nstrumentation). Basis of this bus is the CompactPCI™ bus.

The board can be plugged into any desired slot of a CompactPCI™ or PXI™ system (except for slot 1). It can be definitely identified also in the case that several boards of this type are used in the same rack.

The PCI 3051 communication board is a PC plug-in board for the PCI Local Bus Rev. 2.2.
It can be operated at any PCI slot (32 bits, 33 MHz, 3.3 V)

Both boards do not have jumpers for hardware detection and are automatically integrated into the respective system.

2.2.2 Dimensions

The dimensions of both boards correspond to standard dimensions of the accompanying bus system:

- PXI 3051 CAN Interface Board: 160 mm x 100 mm (L x W)
- PCI 3051 CAN Interface Board: 168 mm x 106 mm (L x W)

2.2.3 PXI 3051/ PCI 3051 Characteristics

Symbol	Parameter	Min.	Typ.	Max.	Unit	Remarks
U_{BAT}	Battery voltage		12	27/ 50	V	Acc. to transceiver's type
	Transmission rate			1	MBaud	
R_{bus}	Terminating resistor 1		120		Ohm	J1..J4 jumpers plugged in
R_{bus}	2 x Terminating resistor 2		5.1		kOhm	on transceiver module
	External trigger input	3.3		50	V	
	External trigger output			U_{BAT}	V	Open collector output via npn transistor

To create a voltage level difference at the external trigger output, an external pull-up resistor must be connected with this output via a voltage source, e.g. 10kΩ via U_{Bat} voltage.

2.3 Construction

2.3.1 General

In the basis version, both boards have two CAN interfaces of version 2.0b. The maximum extension of four CAN interfaces can be achieved by means of set-top boards (Aufsatzboards) and further transceiver modules.

Figure 2-4 shows schematically the construction of the boards in a block diagram.

For the PXI/ PCI 3051 boards, an ASIC is used as the interface to the PCI or cPCI bus. This ASIC includes all the function blocks required for the communication with the computer bus.

The PCI 3051 communication board does not have a PXI interface. To exchange trigger signals with other GOEPEL electronic PCI boards despite of that, an additional plug connector is on this board with two lines configurable as input or output (Trig.con in Figure 2-6).

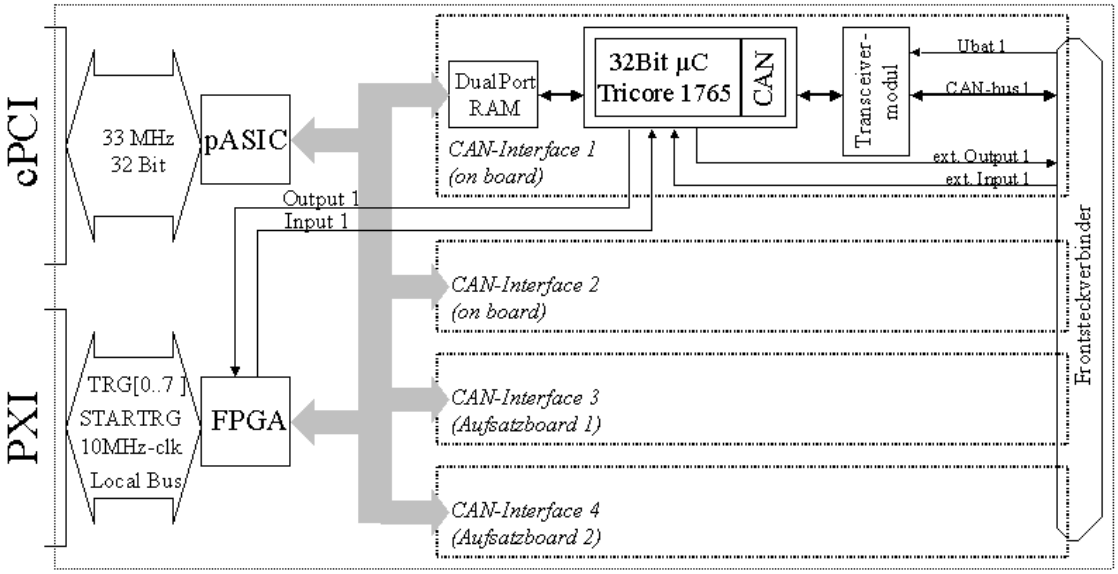

Figure 2-4: Block diagram of a PXI/ PCI 3051 Communication board

2.3.2 Addressing

PXI 3051: PXI racks do have an own geographical slot addressing of the backplane. Numbering starts with 1 and can be seen at the cover's front side. Mount always an embedded controller or an MXI card at slot 1.

The PXI 3051 board can read out this geographical slot address. For that the XILINX has to be loaded with the belonging FPGA file (see XilinxDownload functions for different drivers in the Control Software section).

PCI 3051: PCI racks do not have an own geographical slot addressing. There is a separate address jumper field (Addr.jumper in Figure 2-6) for clear identification of the board (analogously to "geographical addressing" of the PXI specification) in a system with several PCI 3051 boards.You can select up to 16 addressing variants by this. The corresponding binary value (0..15) set with the jumpers can be read out by the delivered software.

2.3.3 Trigger behavior

Each CAN interface has 2 x 2 additional input/ output lines.

One input/ output per CAN interface is connected to the frontal plug connector.

The second input/ output per CAN interface can be connected with the Startrigger and Trigger[0..7] PXI signals or with the two additional I/O lines of the PCI board via the corresponding driver configuration.

The CAN controller firmware can be configured the way that the interface is activated either to the PXI trigger signals or to the external trigger signals.

The following functions are possible:

- ENABLE function:
  an external input signal activates/ deactivates for a CAN interface the possibility of sending messages
- TRIGGER_IN function:
  an external trigger signal actuates the sending of CAN messages prepared before
- TRIGGER_OUT function:
  an output signal is created when sending or receiving a certain message.

2.3.4 Communication Interfaces

4 x CAN Interfaces Version 2.0b at most:

The type of the mounted transceiver is decisive for proper operation of a CAN interface in a network. Often CAN networks do only operate properly in the case that all members use a compatible type of transceiver.

To offer maximal flexibility to the users of the PXI/ PCI 3051 boards, the transceivers are designed as plug-in modules. There are several types (highspeed, lowspeed, single-wire etc.) that can be easily exchanged.

Not only the type of the mounted transceiver, but also the terminating resistor of the bus is very important for proper operation of a CAN network.
For the use of highspeed CAN transceivers, usually one 120 Ohm resistor which is mounted on the board is active for each CAN interface. These resistors can be deactivated by removing the J1..J4 jumpers. Then the resistors can be replaced by inserting wired resistors (to be soldered!) of the desired value at the positions JP1..JP4 (see Figure 2-5 and Figure 2-6).

In the case of lowspeed CAN transceivers, two terminating resistors of 5.1 kOhm each for RTH and RTL per CAN interface are mounted on the transceiver module. Then a wired resistor must not be inserted, and the corresponding jumper has to be removed.

CAN transceivers of the following types require a connection of the battery voltage with the pins 15, 18, 21 or 24 of the XS1 plug connector (V_Bat1..V_Bat4, see Frontal Plug Connector Assignment) for the corresponding CAN interface:

- TJA104A
- TJA1054
- 82C52
- B10011S

2.3.5 Mounting

Figure 2-5 and Figure 2-6 show the component side of the boards schematically. The positions of the optional set-top boards (Aufsatzboard) and the transceiver modules can be seen on this illustration as well as the positions of the J1..J4 jumpers for activating/ deactivating the terminating resistors. A plugged-in jumper means that the 120 Ohm terminating resistor is active.

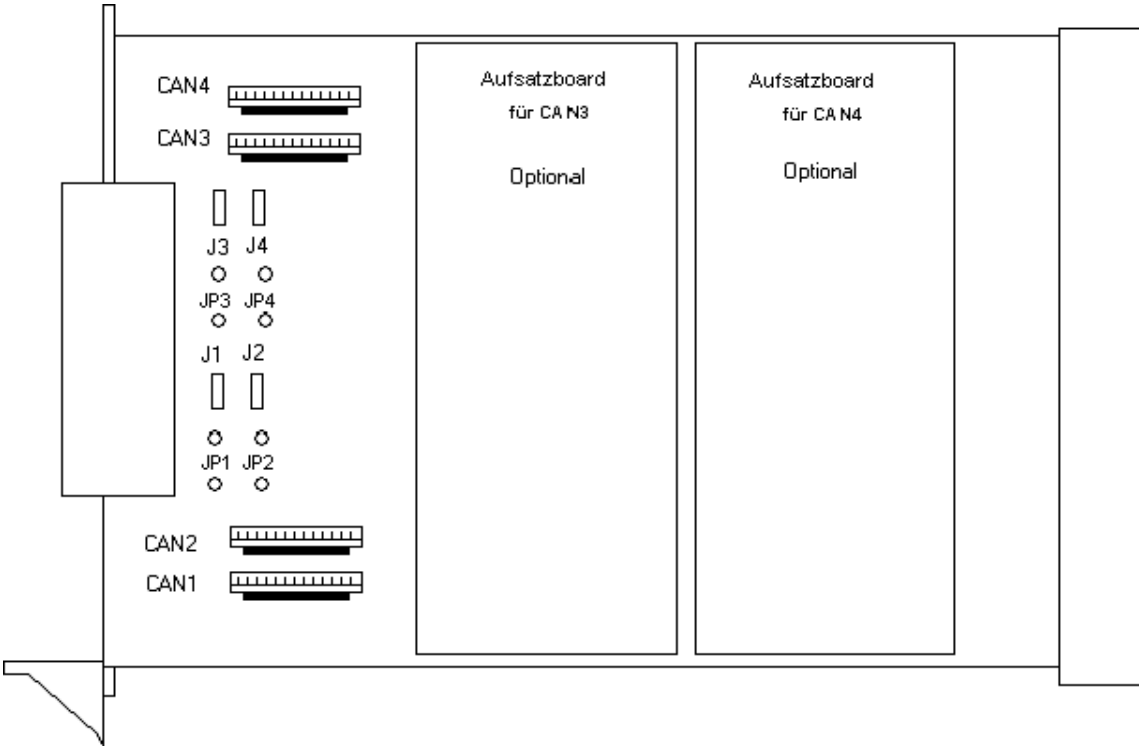

Figure 2-5: PXI 3051 Communication board – Component side (schematically)

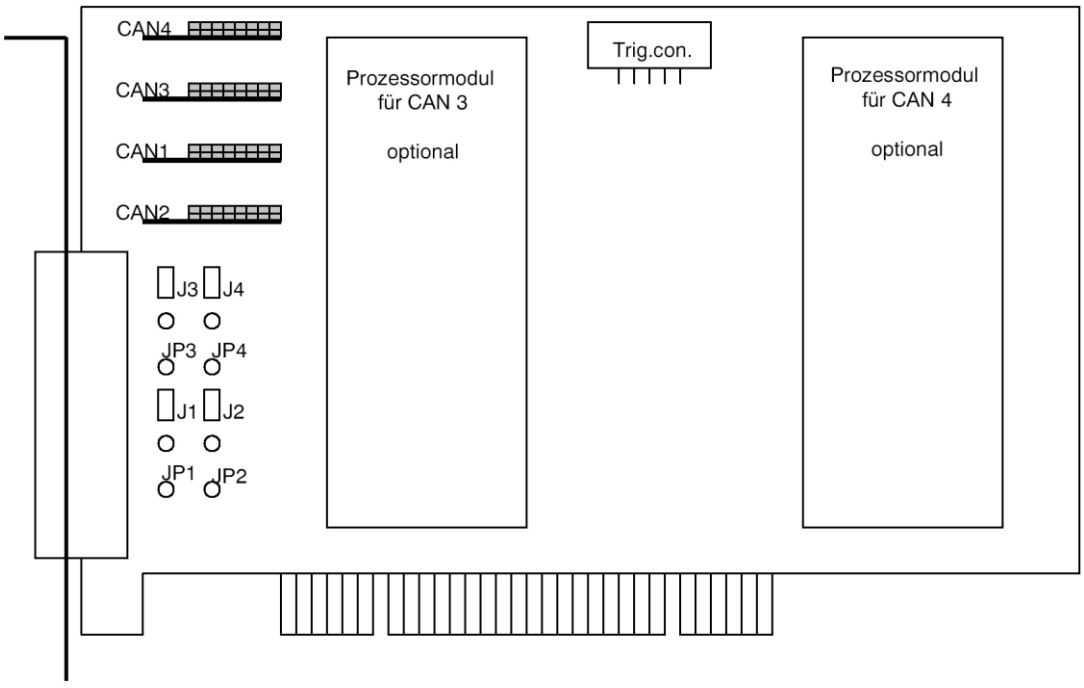

Figure 2-6: PCI 3051 Communication board – Component side (schematically)

The configuration elements of Figure 2-5 and Figure 2-6 are described in the following table:

CAN1	Transceiver module for CAN1
CAN2	Transceiver module for CAN2
CAN3	Transceiver module for CAN3
CAN4	Transceiver module for CAN4
J1	Jumper to activate the **120Ω** bus terminating resistor (onboard) for CAN1
J2	Jumper to activate the **120Ω** bus terminating resistor (onboard) for CAN2
J3	Jumper to activate the **120Ω** bus terminating resistor (onboard) for CAN3
J4	Jumper to activate the **120Ω** bus terminating resistor (onboard) for CAN4
JP1	Position for the optional wired terminating resistor – CAN1
JP2	Position for the optional wired terminating resistor – CAN2
JP3	Position for the optional wired terminating resistor – CAN3
JP4	Position for the optional wired terminating resistor – CAN4
Addr.Jumper	This jumper field on a PCI 3051 board is for clear identification of the board (analogously to "geographical addressing" of the PXI specification) in a system with several PCI 3051 boards. You can select up to 16 addressing variants this way (0..15). The corresponding binary value set with the jumpers can be read out by the delivered software.
Trig.con	Plug connector on a PCI 3051 board to exchange trigger signals with other GOEPEL electronic PCI boards

2.3.6 Frontal Plug Connector Assignment

Type: DSub 25 poles socket

The CAN interfaces are provided via this plug connector at the frontal edge of a PXI/ PCI 3051 communication board.

The assignment of both boards is identical according to the following table:

No.	XS1 pin	Signals name	Remarks
1	14	CAN1_High	CAN high bus connection
2	2	CAN1_Low	CAN low bus connection
3	15	V_Bat1	Transceiver plus reference potential
4	1	GND	Transceiver ground potential
5	17	CAN2_High	CAN high bus connection
6	5	CAN2_Low	CAN low bus connection
7	18	V_Bat2	Transceiver plus reference potential
8	4	GND	Transceiver ground potential
9	20	CAN3_High	CAN high bus connection
10	8	CAN3_Low	CAN low bus connection
11	21	V_Bat3	Transceiver plus reference potential
12	7	GND	Transceiver ground potential
13	23	CAN4_High	CAN high bus connection
14	11	CAN4_Low	CAN low bus connection
15	24	V_Bat4	Transceiver plus reference potential
16	10	GND	Transceiver ground potential
17	3	INPUT1	Input CAN controller 1
18	16	OUTPUT1	Output CAN controller 1
19	6	INPUT2	Input CAN controller 2
20	19	OUTPUT2	Output CAN controller 2
21	9	INPUT3	Input CAN controller 3
22	22	OUTPUT3	Output CAN controller 3
23	12	INPUT4	Input CAN controller 4
24	25	OUTPUT4	Output CAN controller 4
25	13	GND	Ground potential

2.4 Delivery notes

PXI/ PCI 3051 boards are delivered in the following variants:

- 2x CAN Interface
- 3x CAN Interface
- 4x CAN Interface

In addition to selecting an interface, the type of the corresponding CAN transceiver as well as the required Functionalities for each CAN Interface must be selected.

3 Control Software

There are three ways to integrate the PXI 3051/ PCI 3051 hardware in your own applications:

- Programming via G-API
- Programming via DLL Functions
- Programming with LabVIEW

3.1 Programming via G-API

The G_API (GOEPEL-API) is the favored user interface for this GOEPEL hardware.

You can find all necessary information in the *G-API* folder of the delivered CD.

3.2 Programming via DLL Functions

Programming via DLL Functions is possible also in future for existing projects which can not be processed with the GOEPEL electronic programming interface G-API.

We would be pleased to send the GOEPEL Firmware documentation to you on your request. Please get in touch with our sales department in case you need that.

The GPxi3051 and PXI3051 expressions used in the following function description stand for PXI 3051/ PCI 3051.

For the used structures, data types and error codes refer to the headers – you find the corresponding files on the supplied CD.

In this User Manual, Controller ALWAYS means the micro controller assigned to a CAN interface
(with the exception of the “CAN Controller” designation on the front panel for the entire board).

3.2.1 Windows Device Driver

The DLL functions for programming using the Windows device driver are described in the following chapters:

- DriverInfo
- DLL Version
- XILINX – Download
- XILINX – Write Data
- DPRAM – Write Instruction
- DPRAM – Read Response
- DPRAM – Read Monitor
- Reset Port

3.2.1.1 DriverInfo

The GPxi3051_GetDriverInfo function is for the status query of the hardware driver.

Format:

```
int GPxi3051_GetDriverInfo(GPxi3051_StructDriverInfo *pDriverInfo)
```

Parameter:

Pointer, for example pDriverInfo
to a data structure,
For the structure, see the *GPxi3051.h* file on the supplied CD

Description:

The GPxi3051_GetDriverInfo function returns information regarding the status of the hardware driver.

For this reason, the address of a pDriverInfo pointer has to be transferred to the function.
The structure pDriverInfo is pointing to is filled with various information within the function.

3.2.1.2 DLL Version

The GPxi3051_DLL_Version function is for the version number query of the DLL.

Format:

```
int GPxi3051_DLL_Version(unsigned long *pVersion)
```

Parameter

Pointer, for example pVersion
to the Version number

Description:

The GPxi3051_DLL_Version function returns the version number of the *GPxi3051w.dll* as an integer value.

Example:

Version number **1.23** is returned as **123**,
and version number **1.60** as **160**.

3.2.1.3 XILINX – Download

The GPxi3051_XilinxDownload function is to load an FPGA file to the XILINX.

Format:

```
int GPxi3051_XilinxDownload(unsigned long card, char *pFileName)
```

Parameters:

card

Index of the PXI/ PCI 3051 board, beginning left with 1

Pointer, for example pFileName
to the path of the FPGA file to be loaded

Description:

The GPxi3051_XilinxDownload function allows to load an FPGA file (**.cfd* extension) to the XILINX. This file serves, among other possibilities, to read the geographical slot address in the PXI Rack.

The loaded data is volatile. Therefore the function has to be executed again after switching off power.

After XilinxDownload, a delay of about 500 ms is required (as the controllers execute a power-on reset).

Then, carry out the 0x10 Software Reset firmware command for all controllers to come into the normal operating mode from bootloader mode.

3.2.1.4 XILINX – Write Data

The GPxi3051_XilinxWriteData unction allows the configuration and execution of functions provided by the XILINX.

Format:

```
int GPxi3051_XilinxWriteData(unsigned char *data, unsigned long *length)
```

Parameters:

Pointer, for example data
to the Write data area (currently max. 128 bytes per command)

length
Size of the storage area data is pointing to (in bytes)

Description:

Before using the functionality of the XILINX, the corresponding FPGA file must have been loaded by GPxi3051_XilinxDownload (see XILINX – Download).

The data format consists of four bytes including the command.
If necessary parameter bytes can follow.

Data format:
- 1st byte: 0x48 (StartByte)
- 2nd byte: card (index of the PXI/ PCI 3051 board, beginning left with 1)
- 3rd byte: 0x00 (Reserved Byte)
- 4th byte: XILINX command

Currently supported XILINX command:

0x10 PowerOnReset for the complete board

3.2.1.5 DPRAM – Write Instruction

The GPxi3051_DpramWriteInstruction is for sending a command to the selected controller.

Format:

```
int GPxi3051_DpramWriteInstruction(unsigned char *data, unsigned long length)
```

Parameters:

Pointer, for example data
to the Write data area,
consisting of Command Header and Command Bytes
(currently max. 1024 bytes per command)

length

Size of the storage area data is pointing to (in bytes)

Description:

The GPxi3051_DpramWriteInstruction function sends a command to the selected controller.

In the header of the structure data is pointing to, there is the information regarding the PXI/ PCI 3051 board and the belonging controller to be activated by this function.
Therefore these parameters have not to be given separately.

3.2.1.6 DPRAM – Read Response

The GPxi3051_DpramReadResponse function is for reading a response from the selected controller.

Format:

```
int GPxi3051_DpramReadResponse(unsigned long card, unsigned long port,
                               unsigned char *data, unsigned long *length)
```

Parameters:

card

Index of the PXI/ PCI 3051 board, beginning left with 1

port

Number of the controller (1..4)

Pointer, for example data
to the Read data area,
consisting of Response Header and Response Bytes
(currently max. 1024 bytes per response)

length

Value of the parameter before function call:
Size of the buffer pointed by data in bytes

Value of the parameter after function call:
Number of bytes actually read

Description:

The GPxi3051_DpramReadResponse function reads back the oldest response written by the controller (1..4) into the Response area of the DPRAM.

If several responses have been provided by the corresponding controller, but not sent, they are not lost but stored in the form of a list.

On calling up, the GPxi3051_DpramReadResponse function continues to supply data until this list contains no more entries.

3.2.1.7 DPRAM – Read Monitor

The GPxi3051_DpramReadMonitor is for reading the monitor data of the selected controller.

Format:

```
int GPxi3051_DpramReadMonitor(unsigned long card, unsigned long port,
                              unsigned char *data, unsigned long *length)
```

Parameters:

card

Index of the PXI/ PCI 3051 board, left beginning with 1

port

Number of the controller (1..4)

Pointer, for example data
to the Read data area (max. 20kByte)

length

Value of the parameter before function call:
Size of the buffer pointed by data in bytes

Value of the parameter after function call:
Number of monitor entries actually read

Description:

The GPxi3051_DpramReadMonitor function reads the data found in the monitor area of the DPRAM.

This concerns exclusively the data provided by the controller in the Buffer reception monitor Mode. That means, the normal DPRAM Response area is separated from DPRAM's monitor data area (Buffer reception).

20 bytes are required per monitor entry. The length given back is already divided by these 20 bytes and corresponds in this way to the number of monitor entries actually read.

3.2.1.8 Reset Port

The GPxi3051_ResetPort function is for releasing a software reset for the selected controller.

Format

```
int GPxi3051_ResetPort(unsigned long card, unsigned long port)
```

Parameters:

card

Index of the PXI/ PCI 3051 board, beginning left with 1

port

Number of the controller (1..4)

Description:

The GPxi3051_ResetPort function releases a software reset for the selected controller.

This releasing procedure is executed via a separate interrupt channel, NOT via the command interpreter of the software (see 0x10 Software Reset Firmware command).

3.2.2 VISA Device Driver

The DLL functions for programming using the VISA device driver are described in the following sections:

- Init
- Done
- Driver Info
- XILINX – Download
- XILINX – Write Data
- Write Data
- Read Data
- Read Monitor
- Reset Port

3.2.2.1 Init

The PXI3051_Init function is for opening VISA sessions for the system's PXI/ PCI 3051 boards including initialization.

Format:

```
ViStatus PXI3051_Init(ViUInt32 *CardCount)
```

Parameter:

CardCount

Number of the system's PXI/ PCI 3051 boards recognized by the VISA driver.

Description:

The PXI3051_Init function searches for all PXI/ PCI 3051 boards of the system and opens the required sessions.
Additionally, board internal initializations are carried out.

Therefore this function must be executed as the first step.

3.2.2.2 Done

The PXI3051_Done function closes all VISA sessions of the system's PXI/ PCI 3051 boards.

Format:

```
ViStatus PXI3051_Done(void)
```

Parameter:

none

Description:

The PXI3051_Done function closes all VISA sessions of the system's PXI/ PCI 3051 boards.

No further access to the boards is possible then.

3.2.2.3 Driver Info

The PXI3051_DriverInfo function provides general information regarding driver and board.

Format:

```
ViStatus PXI3051_DriverInfo(PXI3051_StructDriverInfo *DriverData,
                                                ViChar *DeviceName)
```

Parameters:

Pointer, for example DriverData
to a data structure
For the structure see the *PXI3051_API.h* file of the supplied CD

DeviceName
`Array[K_DEV_MAX][K_RES_NAME_LENGTH]`
(see *PXI3051_API.h*)

Description:

The PXI3051_DriverInfo function provides information regarding the driver and the system's PXI/ PCI 3051 boards.

The DeviceName indicates the resource names registered by VISA. This information correlates with the display of NI MAX.

3.2.2.4 XILINX – Download

The PXI3051_XilinxDownload function is to load an FPGA file to the XILINX.

Format:

```
ViStatus PXI3051_XilinxDownload(ViUInt32 Card, ViChar *FileName)
```

Parameters:

Card

Index of the PXI/ PCI 3051 board, beginning left with 1

Pointer, for example FileName
to the path of the FPGA file to be loaded

Description:

The PXI3051_XilinxDownload function allows to load an FPGA file (**.cfd* extension) to the XILINX. This file serves, among other possibilities, to read the geographical slot address in the PXI rack.

The loaded data is volatile. Therefore the function has to be executed again after switching off power.

After XilinxDownload, a delay of about 500 ms is required (as all controllers execute a power-on reset).

Then, carry out the 0x10 Software Reset firmware command for all controllers to come into the normal operating mode from bootloader mode.

3.2.2.5 XILINX – Write Data

The PXI3051_XilinxWriteData function allows the configuration and execution of functions provided by the XILINX.

Format:

```
ViStatus PXI3051_XilinxWriteData(ViUInt8 WriteData[])
```

Parameter:

Pointer, for example WriteData
to the Write data area (currently max. 128 bytes per command)

Description:

Before using the functionality of the XILINX, the corresponding FPGA file must have been loaded by PXI3051_XilinxDownload (see XILINX Download).

The data format consists of four bytes including the command.
If necessary parameter bytes can follow

Data format:
- 1st byte: 0x48 (StartByte)
- 2nd byte: card (index of ther PXI/ PCI 3051 board, beginning left with 1)
- 3rd byte: 0x00 (Reserved Byte)
- 4th byte: XILINX command

Currently supported XILINX command:

0x10 PowerOnReset for the complete board

3.2.2.6 Write Data

The PXI3051_WriteData function is for writing data to the selected controller.

Format:

```
ViStatus PXI3051_WriteData(ViUInt8 WriteData[], ViUInt32 Length_In_Bytes)
```

Parameters:

Pointer, for example WriteData
to the Write data area,
consisting of Command Header and Command Bytes
(currently max. 1024 bytes per command)

Length_In_Bytes

Size of the storage area WriteData is pointing to (in bytes)

Description:

The PXI3051_WriteData function allows writing of data to the controller.

In the header of the structure WriteData is pointing to, there is the information regarding the PXI/ PCI 3051 board and the belonging controller to be activated by this function.
Therefore these parameters have not to be given separately.

3.2.2.7 Read Data

The PXI3051_ReadData function is for reading data from the selected controller.

Format:

```
ViStatus PXI3051_ReadData(ViUInt32 Card, ViUInt32 Port,
                          ViUInt8 ReadData[], ViUInt32 *Length)
```

Parameters:

Card

Index of the PXI/ PCI 3051 board, beginning left with 1

Port

Number of the controller (1..4)

Pointer, for example ReadData
to the Read data area,
consisting of Response Header and Response Bytes
(currently max. 1024 bytes per response)

Length

Value of the parameter before function call:
Size of the buffer pointed by ReadData in bytes

Value of the parameter after function call:
Number of bytes actually read

Description:

The PXI3051_ReadData function allows reading of data provided by the controller (see also GPxi3051_DpramReadResponse function in the Windows Device Driver section).

3.2.2.8 Read Monitor

The PXI3051_ReadMonitor function is for reading monitor data from the selected controller.

Format:

```
ViStatus PXI3051_ReadMonitor(ViUInt32 Card, ViUInt32 Port,
                             ViUInt8 MonitorData[], ViUInt32 *Length)
```

Parameters:

Card

Index of the PXI/ PCI 3051 board, beginning left with 1

Port

Number of the controller (1..4)

Pointer, for example MonitorData
to the Read data area (max. 20kByte)

Length

Value of the parameter before function call:
Size of the buffer pointed by MonitorData in bytes

Value of the parameter after function call:
Number of monitor entries actually read

Description:

The PXI3051_ReadMonitor function reads the data found in the monitor area of the controller (see GPxi3051_DpramReadResponse function in the Windows Device Driver section).
This concerns to a separate read area of the board.

3.2.2.9 Reset Port

The PXI3051_ResetPort function is for releasing a software reset for the selected controller.

Format

```
ViStatus PXI3051_ResetPort(ViUInt32 Card, ViUInt32 Port)
```

Parameters:

Card

Index of the PXI/ PCI 3051 board, beginning left with 1

Port

Number of the controller (1..4)

Description:

The PXI3051_ResetPort function releases a software reset for the selected controller.

This releasing procedure is executed via a separate interrupt channel, NOT via the command interpreter of the software (see 0x10 Software Reset Firmware command).

3.3 Programming with LabVIEW

3.3.1 LabVIEW via G-API

The supplied CD contains VIs for activating PXI/ PCI 3051 boards under LabVIEW.

The functions of GOEPEL's G-API are used for this.

3.3.2 LLB using the Windows Device Driver

The supplied CD contains VIs for activating PXI/ PCI 3051 boards under LabVIEW.

The functions described in the Windows Device Driver section are used for this.

3.3.3 LLB using the VISA Device Driver

The supplied CD contains VIs for activating PXI/ PCI 3051 boards under LabVIEW.

The functions described in the VISA Device Driver section are used for this.

3.4 Further GOEPEL Software

PROGRESS, Program Generator and myCAR of GOEPEL electronic GmbH are comfortable software programs for testing with GOEPEL hardware.

Please refer to the corresponding User Manual to get more information regarding these programs.