PXI 6161

MOST150 Interface

User Manual
(Translation of Original docu)
Document Version 1.3

GOEPEL electronic GmbH
Goeschwitzer Str. 58/60 •D-07745 Jena
+ 49-3641-6896-597 • ats_support@goepel.com • www.goepel.com

Printed: 08.01.2015

Issue: January 2015

1 Board Installation

1.1 Hardware Installation

Before beginning with the hardware installation you have to ensure that your system is switched off and disconnected from the mains supply.

Please refer also to the user manual of your PXI system for additional installation instructions that possibly have to be followed.

Electro Static Discharge (ESD) can harm your system and destroy electronic components. This can lead to irreparable damage on both the PXI 6161 controller board and the system hosting the board as well as to unexpected malfunction of your test system.
Therefore do not touch the board surface or any connector pins and electronic components.

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 fixation screws if necessary and remove the cover from the slot.

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

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 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 Installation

PXI 6161 boards can be operated under Windows® 2000/ XP as well as under Windows®7/ 32 bit and Windows®7/ 64 bit.

Due to the plug and play capability of Windows®, 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 in the *Series61xx* folder of the supplied CD.

If necessary, you can find the required *inf* files as follows:

- *Pxi61xx.inf* for Windows® 2000/ XP in the *Driver PXI_PCI - W2K, WinXP (Version xx)* folder
- *Pxi61xx.inf* for Windows®7/ 32 bit in the *Driver PXI_PCI - Win7_x32 (Version xx)* folder
- *Pxi61xx_x64.inf* for Windows®7/ 64 bit in the *Driver PXI_PCI - Win7_x64 (Version xx)* folder

It is not absolutely essential to restart the system.

The following step is only required in case you do not use the G-API (see also Programming via G-API).

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.

1.2.2 VISA Device Driver Installation

1st Step

Copy the
VISA_Driver PXI_PCI - W2K, WinXP (Version xx) and
VISA_Driver PXI_PCI - Win7_x32_x64 (Version xx) folder
from the *Series61xx* folder of the delivered CD to your hard disk
(recommendation: Copy the complete folders to *C:* \).

2nd Step

VISA for Windows 2000® / XP

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 and enter as target directory the one which contains the *PXI61xx.inf* file.
(according to recommendation:
C:*VISA_Driver PXI_PCI - W2K, WinXP (Version xx)*.

VISA for Windows® 7

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 and enter as target directory the one containing the *PXI61xx_VISA.inf* file.
(according to recommendation:
C:*VISA_Driver PXI_PCI - Win7_x32_x64 (Version xx)*.

VISA for LabView RT

For operating PXI 6161 boards under the RT operating system, use the *PXI61xx.inf* file from the folder mentioned above.

Copy this file to the *ni-rt**system* folder of the embedded controller.
Use the NI Measurement & Automation Explorer for that.
The connected RT controller can be found under Remote Systems.
Open a pop-up menu by pressing the right mouse button.
In this menu, select the File Transfer entry and follow the instructions.

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

3rd Step

Reboot your computer to complete installation.

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

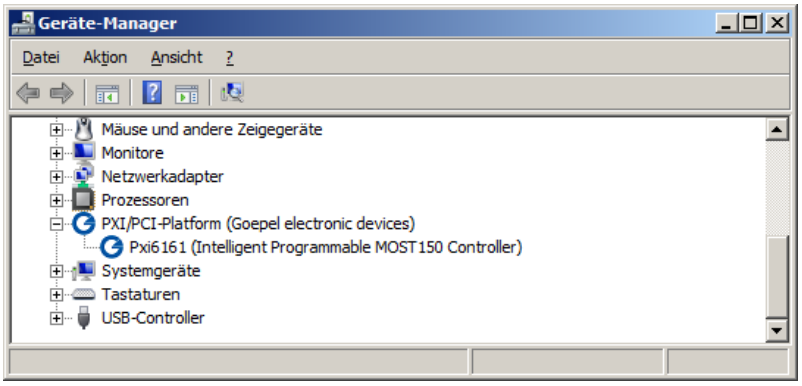

Figure 1-1: Windows

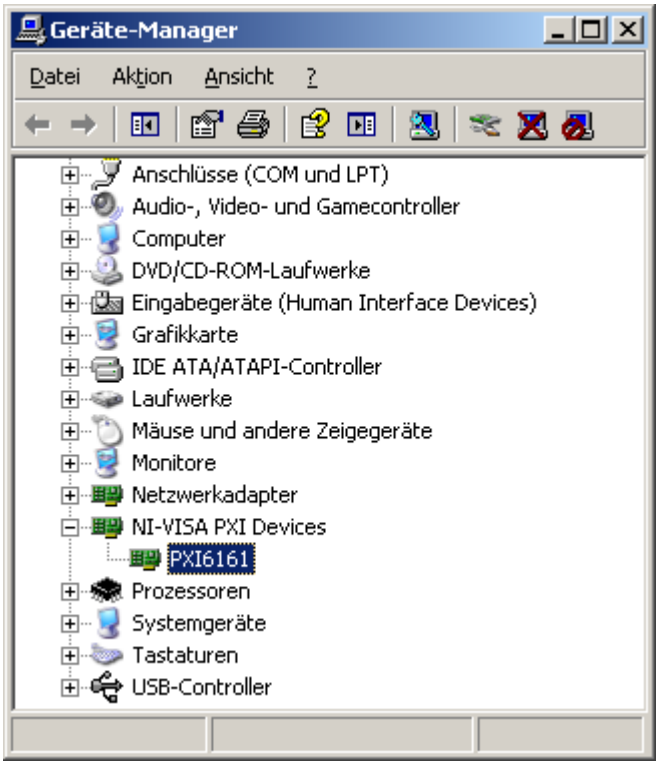

Figure 1-2: VISA for Windows XP

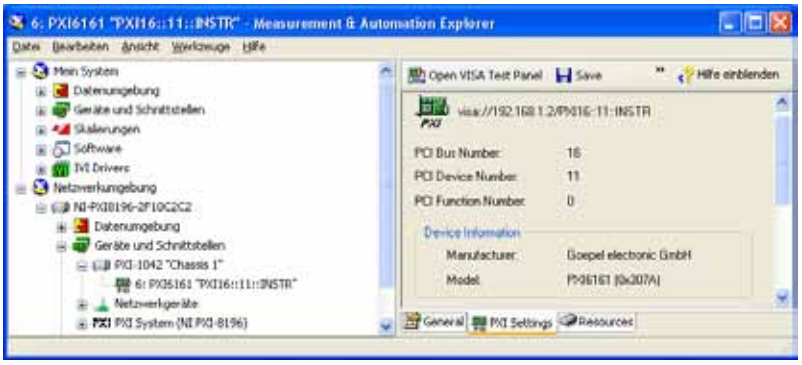

Figure 1-3: VISA for LabVIEW RT

1.2.3 Ethernet

If the Ethernet interface is used for communication with the control PC, there is no driver installation required. The device can be directly addressed via the IP Address (see also Addressing).
This IP Address can be changed by the HardwareExplorer.
The newly set IP Address becomes effective after a restart.

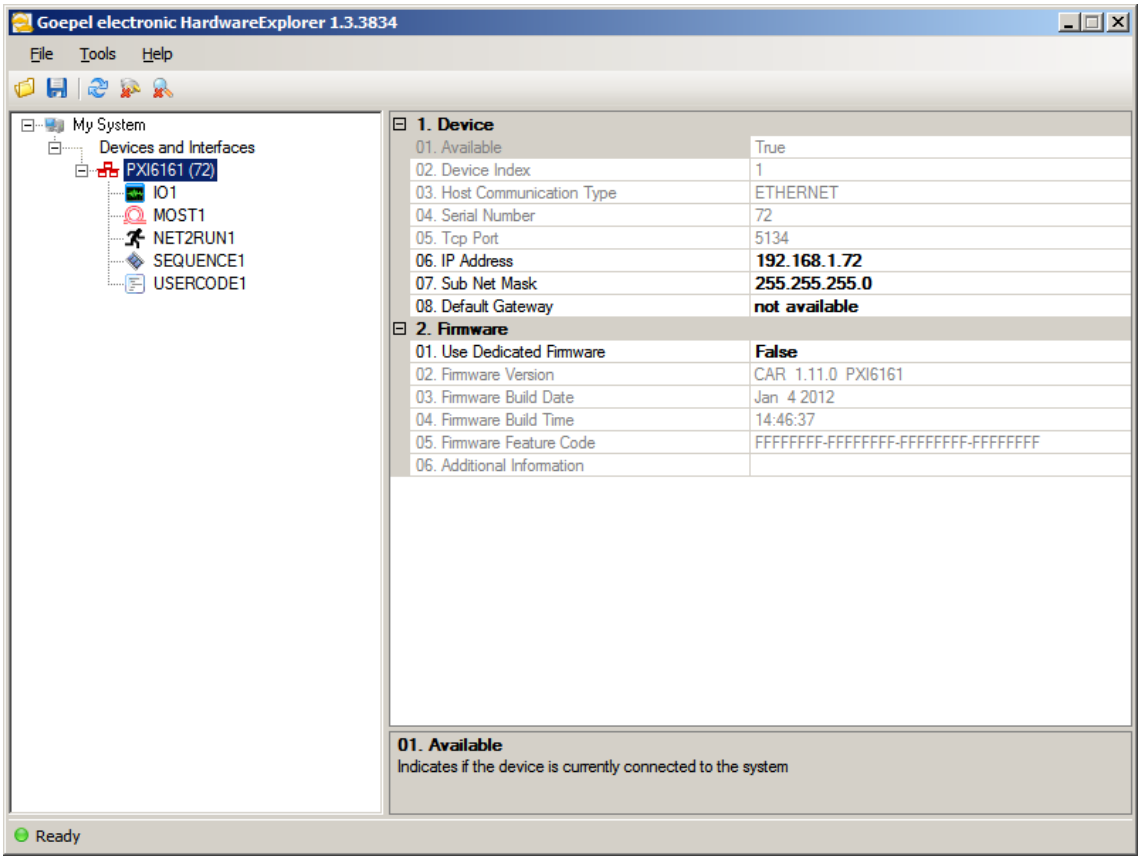

Figure 1-4: IP Address in the GOEPEL electronic HardwareExplorer

2 Hardware

2.1 Definition

The PXI 6161 board is a programmable, intelligent MOST150 controller of GOEPEL electronic GmbH for multimedia applications in automotive and consumer electronics. Optionally it can be extended by additional communication interfaces.

Each PXI 6161 board offers the following resources:

- 1 optical MOST150 interface with a data rate up to 150Mbit/s
- 3 operation modes: Master, Slave, Bypass
- Spy function: Possibility of monitoring the MOST data as passive bus member (board in the Bypass mode) OR as active bus member (board in the Master or Slave mode)
- Independent timer onboard with time stamp resolution greater than or equal to 8 ns
- The master frame rate can be changed between 44.1 kHz and 48 kHz
- optionally 1 DVI output, 1 S/PDIF input/ 1 S/PDIF output
- Ring break dignosis function via frontal plug connector
- 4 digital inputs and 4 digital outputs at the frontal connector (e.g. for extended trigger functions)
- optionally 2 CAN/ LIN/ K-Line interfaces
- 600MHz Power PC with 512MB RAM, 256MB Flash
- The Communication and I/O interfaces are galvanically separated from the user interface (but **not** the resources of the extension board!)
- High fexibility by pluggable transceiver modules and possible variants of the extension board
- 1 GBit Ethernet connector at the front panel as volume data and debug interface
- Visualisation of the controller states by four LEDs arranged at the front panel (see LED Display)

The following picture shows the basic PXI 6161 board
(the MOST interface is covered by the yellow protective cap):

Figure 2-1: PXI 6161

2.2 Technical Data

2.2.1 General

The PXI 6161 controller board is a slot-in board developed for the PXI™ bus.
PCI e**X**tensions for **I**nstrumentation (PXI™) is a modular instrumentation platform originally introduced in 1997 by National Instruments and now promoted by PXI Systems Alliance (PXISA).

PXI™ is based on the CompactPCI™ bus, and it offers all of the benefits of the PCI architecture including performance, industry adoption and COTS technology.
PXI provides a rugged CompactPCI mechanical form-factor, and is standardized by an industry consortium that defines hardware, electrical, software, power and cooling requirements, leaving nothing to chance.

Then PXI™ adds integrated timing and synchronization that is used to route synchronization clocks, and triggers internally. PXI™ is a future-proof technology, and is designed to be simply and quickly reprogrammed as test, measurement, and automation requirements change.

The PXI 6161 controller board operates as PXI slave, therefore the board may be plugged into any desired slot of a PXI chassis except slot 1. PCI Plug & Play auto detection mechanism is supported by the PXI 6161 controller board. No jumper configuration is needed for PXI integration.

CompactPCI and PXI products are interchangeable, that means they can be used in either CompactPCI or PXI chassis; but installation in the alternate chassis type may eliminate certain clocking and triggering features.

So for example you could mount a CompactPCI network interface controller in a PXI rack to provide additional network interface functions to a test stand. Conversely, a PXI module installed in a CompactPCI chassis would not utilize the additional clocking and triggering features of the PXI module.

The PXI 61xx controller board is already prepared for use in a PXI Express hybrid slot, which delivers support for both the PCI and the PCI Express bus by taking advantage of available pins on the high-density backplanes.

2.2.2 Dimensions

A PXI 6161 controller board ist ein 3U standard PXI module covering one slot width.
Board dimensions without front panel and holder:

- PXI 6161: 160 mm x 100 mm (L x W)

2.2.3 PXI 6161 Characteristics

A PXI 6161 board has the following characteristics:

Symbol	Kennwert	Min.	Typ.	Max.	Unit	Remarks
Optical MOST150 Interface						
N	Number		1			
C	Transmission rate			150	MBit/s	
Ring break diagnosis interface						Reference potential GNDiso
N	Number		1			
U_O	Output voltage	1		UBAT	V	
$UBAT_{internRBD}$	Internal battery voltage		12		V	detachable
$UBAT_{externRBD}$	External battery voltage		12	27	V	
Digital inputs						Reference potential GNDiso
N	Number		4			
U_{IH}	High level input voltage	3.5		5.5	V	
U_{IL}	Low level input voltage			1.5	V	
I_L	Input leakage current			35	µA	
Digital outputs						Reference potential GNDiso
N	Number		4			
U_{OH}	High level output voltage	4.8		5	V	
U_{OL}	Low level output voltage			0.5	V	
I_{OUT}	Output current			8	mA	
Additional interfaces						
	CAN/ LIN/ KLine Interfaces			2		See OnBoard Interfaces
	AV Resources Extension			1		See AV Extension

2.3 Construction

2.3.1 General

The core of the PXI 6161 MOST150 board builds a strong 600MHz AMCC 460EX PowerPC. This dual-issue, superscalar 32-bit RISC CPU is based on the Book-E enhanced PowerPC architecture. With features including out-of-order execution, dynamic branch prediction and a highly pipelined double precise floating-point unit, this processor provides the calculation power required for processing complex residual bus simulation on multiple bus interfaces. Furthermore the board comes equipped with a 512MB fast 400MHz DDR2 RAM and 256MB Flash memory of which over 80% is available for user programs.

The PXI 6161 controller board has been designed as highly flexible MOST150 controller platform. It provides an optical MOST150 interface for the transmission of asynchronous, isochronous, control and MOST Ethernet data packets.

Three different operation modes are supported:

- Master (active bus member)
- Slave (active bus member)
- Bypass (passive bus member)

In any operation mode, the monitoring of received or sent MOST150 data is possible.

The following schematic figure shows the distribution of the MOST signal:

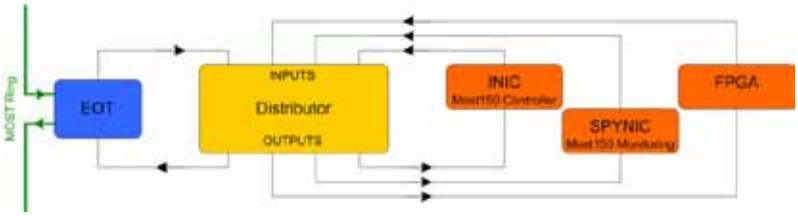

Figure 2-2: MOST Signal distribution

The board is included by the electrical-optical transceiver (EOT) into the MOST ring.

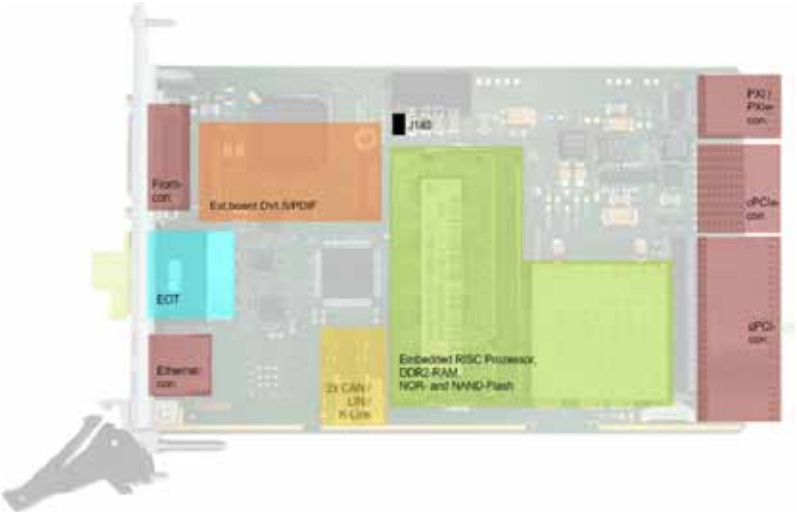

Figure 2-3: PXI 6161 schematically

Below the electrical-optical transceiver (EOT) there is in Figure 2-3 a socket providing access to the 1 GBit Ethernet interface.
It is used for device control, as debug interface or to transfer large data amounts (e.g. monitor data).

Additionally, the PXI 6161 board offers optionally two further serial bus nodes, each of them configurable as CAN, LIN or KLine interface. A transceiver slot belongs to each node. The plugged-in transceiver determines the type of the corresponding node (see also OnBoard Interfaces).
These bus nodes are conducted to the 50-pole frontal connector (same as all other interfaces like ring break diagnosis, audio and video outputs) as well as the digital IOs.

Above the frontal connector, there are four LEDs showing the operating state of the PXI 6161 board (see LED Display).

2.3.2 Isolation

Electric surges can harm expensive test equipment and lead to unreliable test results. Electric isolation protects against electric surges and can help to suppress dangerous electrical transients. It also eliminates ground loops, responsible for data errors due to ground potential differences.

The PXI 6161 controller board provides electric isolation between the PXI system and all the IO signals available at the front connector. This includes the CAN, LIN and MOST communication interfaces as well as digital and analog IOs.

The system requires possibly a ground reference between the GNDiso potential on the frontal connector and the ground potential of the device under test (ECU, etc.).

Jumper J140 (see Figure 2-3:
PXI 6161 schematically) provides a means of introducing a ground reference between the GNDiso potential on the frontal connector and the ground potential of the PXI system.

Making a ground connection to the PXI system may result in high current flow over the test leads and the PXI 6161 controller board.
This could lead to malfunction, wrong test results as well as damage the controller board or other test equipment.
Before closing jumper J140, you must ensure that the device under test and all other devices connected to the frontal connector of the controller board are only supplied via an isolated power supply!

2.3.3 MOST Interface

For proper operation of a MOST interface in a network, all partners must communicate with the same system clock. This depends, among others, on the Master frame rate of the system.

USB 6161 boards support the following Master frame rates:

- 48 kHz and
- 44.1 kHz

(G-API command G_Most_Node_SetProperties, parameter ClockMode)

2.3.4 Addressing

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 6161 board can read out this geographical slot address.

In case of using the Ethernet interface, the device can be controlled via the default IP Address 192.168.1.62, which can be changed if required (see also Driver Installation/ Ethernet).
In principle, there are two ways for this:

- HardwareExplorer: Select the device, under Device set the required IP Address; the new IP Address is effective after restart
- G API Command G_Common_Ethernet_IpAddress_Set; the new IP Address is effective after restart

2.3.5 LED Display

The LEDs arranged at the front panel indicate the current operation state of the microcontroller of the MOST interface.

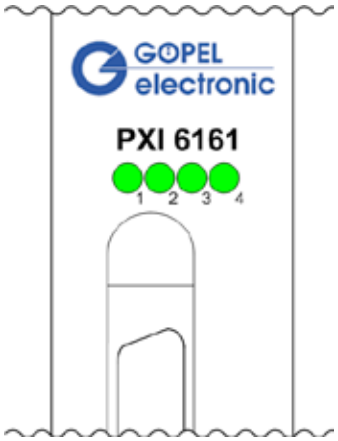

Figure 2-4: LED Display

The following table shows important display states of these LEDs:

State				Remarks
LED 1	**LED 2**	**LED 3**	**LED 4**	
alternately blinking				Bootloader software runs Error reason (probably): Software reset not effected
	blinking			Firmware runs
ON (shortly)				Display while a Firmware command is executed
			ON	Ethernet connection established

This LED display is effected with low priority and can be affected by other running programs.

2.3.6 Connector Pinout

Type: Samtec VRDPC-50-01-M-RA
Required connecting cable: Samtec VPSTP-16-1000-01

The pinout of the frontal connector is shown in the following table:

Pin	*Signal*				*Pin*	*Signal*		
1	*GND*				26	*GND*		
2	**CAN1_H**	**LIN1**	**K-Line1**		**27**	**CAN2_H**	**LIN2**	**K-Line2**
3	**CAN1_L**		**L-Line1**		**28**	**CAN2_L**		**L-Line2**
4	*GND*				29	*GND*		
5	$UBAT_{externCAN1}$				30	$UBAT_{externCAN2}$		
6	GNDiso				31	GNDiso		
7	*GND*				32	*GND*		
8	DIGITAL_OUT1				33	DIGITAL_IN1		
9	DIGITAL_OUT2				34	DIGITAL_IN2		
10	*GND*				35	*GND*		
11	DIGITAL_OUT3				36	DIGITAL_IN3		
12	DIGITAL_OUT4				37	DIGITAL_IN4		
13	*GND*				38	*GND*		
14	**R**ing**B**reak**D**iagnosis				39	GNDiso		
15	$UBAT_{externRBD}$				**40**	N.C. | **DVI.+5Vout**		
16	*GND*				41	*GND*		
17	N.C. | **S/PDIF in**				**42**	N.C. | **DVI.SDA**		
18	N.C. | **S/PDIF out**				**43**	N.C. | **DVI.SCL**		
19	*GND*				44	*GND*		
20	N.C. | **DVI.TX0_p**				**45**	N.C. | **DVI.TX2_p**		
21	N.C. | **DVI.TX0_n**				**46**	N.C. | **DVI.TX2_n**		
22	*GND*				47	*GND*		
23	N.C. | **DVI.TX1_p**				**48**	N.C. | **DVI.TXC_p**		
24	N.C. | **DVI.TX1_n**				**49**	N.C. | **DVI.TXC_n**		
25	*GND*				50	*GND*		

The pinout of the **communication interfaces** depends on the plugged-in transceiver (see OnBoard Interfaces), while the **Pins 17..24** and **40..49** remain empty (but not the GND pins), OR, with plugged in AV Extension board 1, have the pinout according to the table above (see also AV Extension).

2.3.7 OnBoard Interfaces

A PXI 6161 controller board is providing two further communication interfaces, either remaining empty or optionally designed as CAN, LIN or K-Line interface. When required, the assignment can simply be changed by changing the belonging transceiver.

Please proceed extremely carefully when changing transceivers, and pay attention to their position and orientation.

The position of the transceivers and their assignment to the interface number is shown in the following figure:

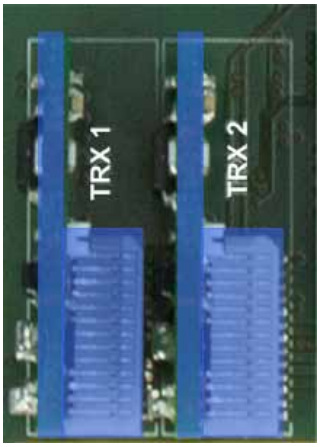

Figure 2-5: Transceiver positions

Each module type is coded and can be identified clearly.
At present, the following types of transceivers are available:

- TJA1041A – CAN (high-speed)
- PCA82C251 – CAN (high-speed)
- TJA1054 – CAN (low-speed)
- B10011S – CAN (truck and trailer)
- TJA 1020 – LIN
- L9637D – K-Line

Both interfaces are generally supplied by an internal voltage of 12V ($UBAT_{intern}$). In case of using other voltages, this internal voltage can be switched off individually by software.

(G-API commands
G_Can_Node_InternalVBat_Disable or
G_Lin_Node_InternalVBat_Disable or
G_KLine_Node_InternalVBat_Disable)

Then, an external voltage ($UBAT_{extern}$) must be supplied via the predefined pins of the frontal connector.

Both communication interfaces, including the belonging $UBAT_{extern}$ inputs, use the GNDiso pins as reference potential.

In case the internal power supply should be later used again, proceed the G_Can_Node_InternalVBat_Enable, G_Lin_Node_InternalVBat_Enable or G_Kline_Node_InternalVBat_Enable G-API commands.

The following table shows the assignment if the communication interfaces:

ID	Interface	Node
1	CAN 1	TRX1
2	CAN 2	TRX2
3	Reserved	
4	Reserved	
5	LIN 1	TRX1
6	LIN 2	TRX2
7	Reserved	
8	Reserved	
9	K-LINE 1	TRX1
10	K-LINE 2	TRX2

Symbol	Identification	Min.	Typ.	Max.	Unit	Bemerkung
CAN V2.0B Interfaces						GNDiso ref. potential
C	Transmission rate			1	Mbit/s	
$UBAT_{intern}$	Internal battery voltage		12		V	detachable
$UBAT_{extern}$	External battery voltage			27	V	
RCAN	Termination high-speed transceiver		120		Ω	detachable
RCAN	Termination low-speed transceiver		10		kΩ	
LIN V2.1 Interfaces						GNDiso ref. potential
C	Transmission rate			19.2	kbit/s	
$UBAT_{internCAN}$	Internal battery voltage		12		V	detachable
$UBAT_{externCAN}$	External battery voltage		12	27	V	
RLIN	Pullup resistor	1	30		kΩ	switchable Master/Slave
K-Line Interfaces						GNDiso ref. potential
C	Transmission rate			9.6	kbit/s	
$UBAT_{intern}$	Internal battery voltage		12		V	detachable
$UBAT_{extern}$	External battery voltage		12	27	V	

Notes on R_{CAN} for the high speed transceiver:
The 120Ω bus termination can be deactivated by software (G-API command G_CAN_Node_BusTermination_Disable, reactivation by G_CAN_Node_BusTermination_Enable).

Notes on RLIN: The 1kΩ pullup resistor corresponds to the LIN Master bus termination and can be activated by software (G-API command G_Lin_PullUpResistor_Enable à Master, Deactivation by G_Lin_PullUpResistor_Disable à Slave).
If it is not active, the internal termination resistance of the LIN transceiver becomes effective (typically 30kΩ for TJA1020).

2.3.8 AV Extension

Plugging-in an AV Extension board provides additional interfaces for audio/ video signals.

At present, GOEPEL electronic GmbH offers the Type 1 AV Extension board with the following characteristics:

Symbol	Characteristic	Min.	Typ.	Max.	Unit	Remarks
DVI D Output	DVI 1.0 compliant					GND ref. potential
N_{LINK}	Number of links		1			Single link
C_{DVI}	Transmission rate	25		48	MBit/s	
R	Resolution	640x420		1600x1200	Pixel	
S/PDIF Audio output	IEC60958-3 compliant					GND ref. potential
$C_{SPDIF\ OUT}$	Transmission rate	32		48	k Frames/s	
S/PDIF Audio input	EC60958-3 compliant					GND ref. potential
$C_{SPDIF\ IN}$	Transmission rate	32		48	k Frames/s	

All pins of the Type 1 AV Extension board use the GND pins as reference potential.
This way, the GND potential is connected to the ground potential of the PXI rack.

2.4 Product Information

The PXI 6161 as intelligent, programmable MOST150 controller forms a highly flexible, adaptable controller platform that can be combined with a multitude of hardware and software recources.

Please refer to the following list of the available options:

Variant	Description
Options for PXI 6161 Controller boards	
CAN Node	Additional CAN node to upgrade on 1 or 2 communication nodes, incl. transceiver module Note: The total quantity of installable CAN/ LIN/ K-Line nodes at the same time amounts 2 per board
LIN Node	Additional LIN node to upgrade on 1 or 2 communication nodes, incl. transceiver module Note: The total quantity of installable CAN/ LIN/ K-Line nodes at the same time amounts 2 per board
K-Line Node	Additional K-Line node to upgrade on 1 or 2 communication nodes, incl. transceiver module Note: The total quantity of installable CAN/ LIN/ K-Line nodes at the same time amounts 2 per board
AV Module Type 1	AV Extension module Type 1 incl. 1 electrical S/PDIF input/ 1 electrical S/PDIF output and 1 DVI output Note: The total quantity of installable AV Extension modules at the same time amounts 1 per board This option is useable independent from and additional to options CAN/ LIN/ K-Line
CAN TJA1054	CAN low speed transceiver module type TJA 1054
CAN TJA1041A	CAN high speed transceiver module type TJA 1041A
CAN NCV7356D1G	CAN single wire transceiver module type NCV7356D1G
LIN TJA1020	LIN Transceiver module type TJA 1020
K-Line L9637D	K-Line Transceiver module type L9637D
DIAG KW2000 TP1.6	Keyword 2000 on TP1.6 on-board CAN diagnostic software
DIAG KW2000 TP2.0	Keyword 2000 on TP2.0 on-board CAN diagnostic software
DIAG KW2000 ISO-TP	Keyword 2000 on CAN-ISO-TP on-board CAN diagnostic software
DIAG UDS ISO-TP	UDS on CAN-ISO-TP on board CAN dagnostic software
DIAG GMLan	GMLan on-board CAN diagnostic software
DIAG J1939	J1939 on-board CAN diagnostic software
CAL CCP2.1	CAN Calibration protocol CCP2.1
LIN adv-lib	Advanced library for test of the LIN protocol specific. 2.0/ 2.1
CAN DBC	Automated read-in of the CAN data base (DBC file) (LabVIEW based)
LIN LDF	Automated read-in of the LIN data base (LDF file) (LabVIEW based)
Net2Run IDE	Software programming environment (Windows host) to create G-API based onboard UserCode programs on PXI 6161 boards, includes: Net2Run IDE, QNX Neutrino CLT, G-API onboard libraries, single developer license
UserCode Runtime	UserCode runtime module for the execution of G-API based onboard UserCode programs on PXI 6161 boards (this option is necessary for each PXI 6161 board)

3 Control Software

There are several ways to integrate PXI 6161 hardware in your own applications:

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

3.1 Programming via G-API

The G-API (GOEPEL-API) is a programming environment based on “C” for GOEPEL electronic hardware under Windows®. So the G-API is the preferred programming environment for this hardware. It provides a wide, hardware independent command set for CAN, LIN, K-Line, FlexRay, MOST, LVDS, ADIO and Diagnostic services. No matter whether a PXI/ PCI, USB and Ethernet device is used, the commands remain the same.

The hardware abstraction introduced with the G-API gives the test application parallel access to the hardware, allowing one application to access multiple hardware interfaces, as well as multiple applications can access the same hardware interface in parallel.

Another feature introduced by the G-API is the asynchronous hardware access. This means no execution blocking for pending firmware commands. The command acknowledgement is provided via a callback mechanism.

With the Hardware Explorer GOEPEL electronic provides an efficient hardware configuration and management tool, offering users an easy way to manage their hardware configurations and identifying specific hardware interfaces by logical names. Using logical interface names in the application saves from rebuilding the application when porting it to another interface or controller board, as the interface can be easily reassigned in the Hardware Explorer.
Furthermore, the Hardware Explorer provides a simple means of testing the interaction between hardware and software by executing the integrated self-tests.

The figure below shows the GOEPEL electronic Hardware Explorer:

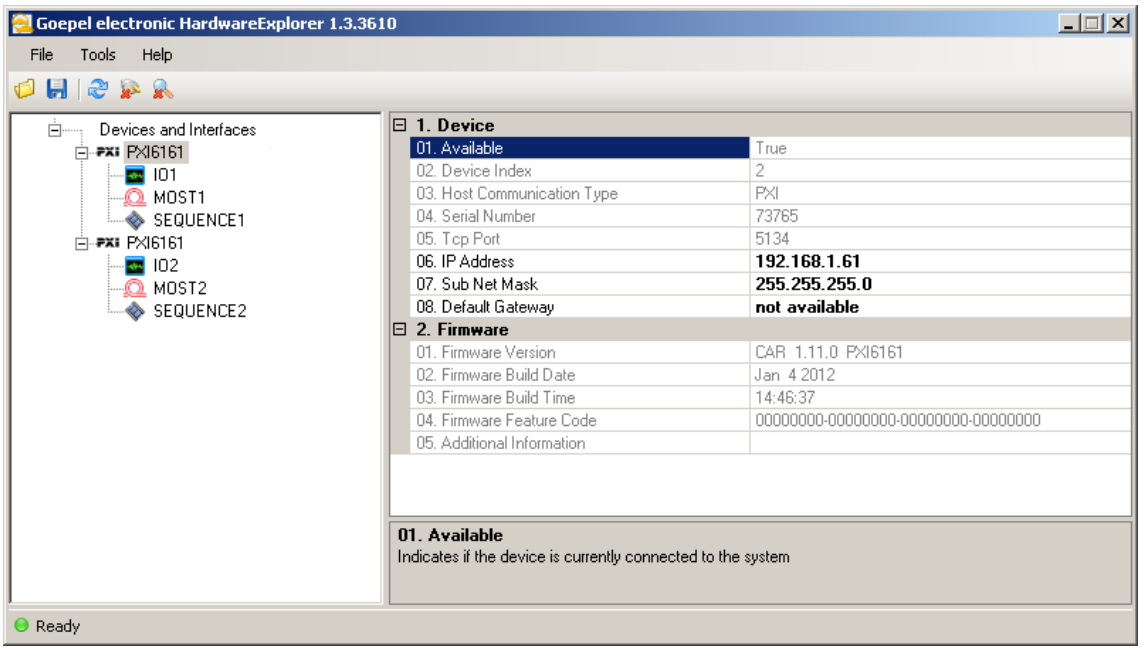

Figure 3-1: Hardware Explorer

Please consult the G-API documentation for further information. This documentation and the installation software are located in the *G-API* folder of the supplied “Product Information” CD.

3.2 UserCode Programming

The PXI 6161 controller board can execute user programs direct on its PowerPC processor. This requires the UserCode run-time module being enabled.

The UserCode run-time module is an option for PXI 6161 boards and requires one license per controller board.

Executing programs directly on the PowerPC improves the real-time performance remarkable and frees up PCI bandwidth on the host system.
Therefore GOEPEL has ported and enhanced their C-programming user interface called G-API from Windows® to the QNX Neutrino real-time operating system. The QNX Neutrino real-time operating system is based on a micro kernel architecture, providing clear separation between the kernel and each individual application. This allows user applications to run in a separate memory space, which ensures safe test execution and improves reliability.

The onboard G-API uses a superset of the G-API commands for Windows® ensuring an easy migration of existing code. Additional functions will provide access to event notifications, timer tasks, the FLASH file system and other RTOS resources as well as standard C libraries.
The PowerPC uses big-endian byte order which must be taken care of when writing or porting code for the onboard G-API. For smooth migration from little to big-endian, a library of conversion macros is provided with the Net2Run IDE development system.

With the Net2Run IDE development system, GOEPEL electronic provides a complete tool chain for creating UserCode programs and for their direct execution on the PXI 6161 controller board.

The Net2Run IDE development system is based on Eclipse IDE and contains the QNX Neutrino Command Line Tools (CLT), including PowerPC-Compiler, Linker and Debugger.
UserCode programs can be downloaded and debugged direct from Net2Run IDE via an Ethernet connection.

The following figure shows the Net2Run IDE:

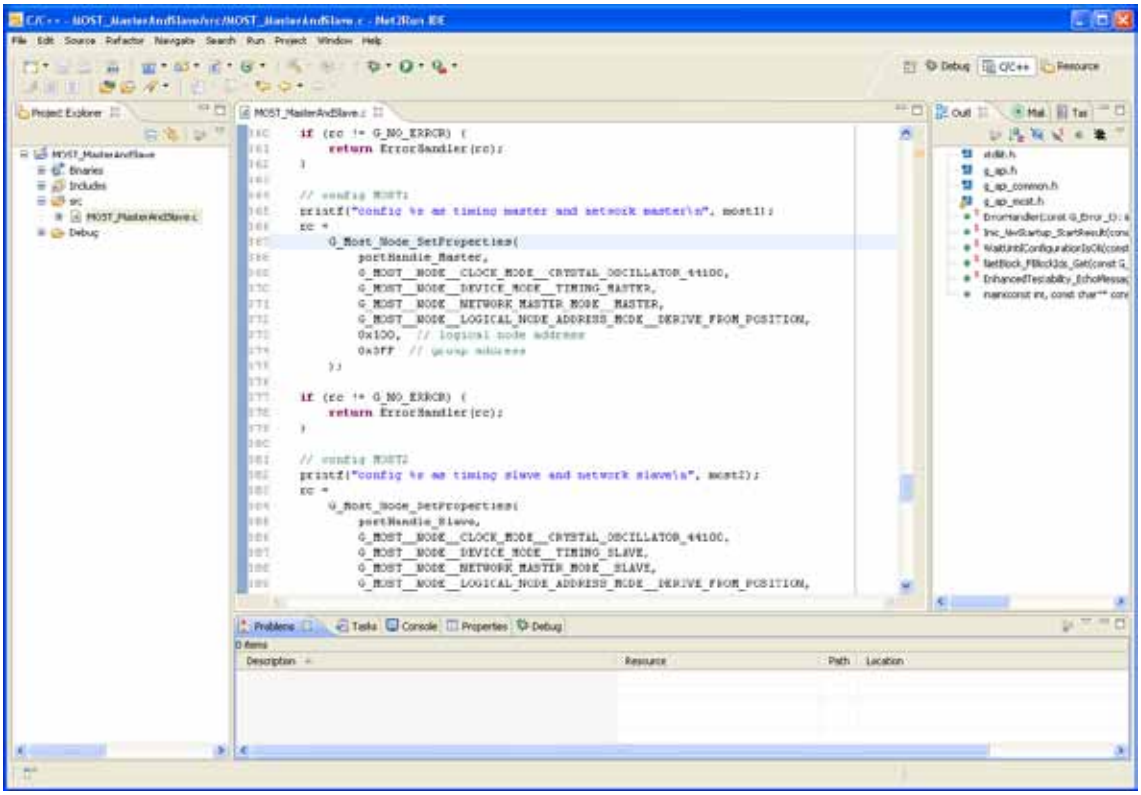

Figure 3-2: Net2Run IDE Development system

Please consult the G-API documentation for further information. This documentation and the installation software are located in the *G-API* folder of the “Product Information” CD.

3.3 Programming via DLL Functions

Programming via DLL Functions is also in future possible e.g. 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.

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

3.3.1 Windows Device Driver

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

- System Info
- Transceiver Info
- Write Instruction
- Read Response
- Read Response Block

3.3.1.1 System Info

The Pxi61xx__SystemInfo function is for the status query of the hardware driver and the board characteristics.

Format:

```
S32 Pxi61xx__SystemInfo(t_System_Info *pSystemInfo,
                                  U32  LengthInByte);
```

Parameters:

Pointer, for example pSystemInfo, to a data structure

(For the structure, see the *Pxi6100_UserInterface.h* file on the supplied CD)

LengthInByte

Size of the memory area pSystemInfo is pointing to, in bytes

Description:

The Pxi61xx__SystemInfo function returns information regarding the status of the hardware driver.

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

3.3.1.2 Transceiver Info

The Pxi61xx__TransceiverInfo function provides information regarding the plugged-in transceivers as well as their number.

Format:

```
S32 Pxi61xx__TransceiverInfo(t_Transceiver_Properties *pTransceiverProperties,
                                                      U32  LengthInByte);
```

Parameters:

Pointer, for example pTransceiverProperties, to a data structure

(For the structure, see the *Pxi6100_UserInterface.h* file on the supplied CD)

LengthInByte

Size of the memory area pTransceiverProperties is pointing to, in bytes

Description:

The Pxi61xx__TransceiverInfo function returns information regarding the transceiver characteristics.

For this, the address of a pTransceiverProperties pointer must be transferred to the function.
Within the function, the structure pTransceiverProperties is pointing to is filled with various pieces of information.

3.3.1.3 Write Instruction

The Pxi61xx__WriteInstruction function is for sending a command to the PXI 6161 controller.

Format:

```
S32 Pxi61xx__WriteInstruction(U8 *pData,
                              U16  DataLength);
```

Parameters:

Pointer, for example pData, to the Writing data area, consisting of Command header and Command bytes (currently max. 1024 bytes per command)

DataLength

Size of the memory area pData is pointing to, in bytes

Description:

The Pxi6161__WriteInstruction function sends a command to the PXI 6161 controller.

In the header of the structure pData is ponting to, there is the information regarding the PXI 6161 board to be activated. Therefore this parameter has not to be given separately.

3.3.1.4 Read Response

The Pxi61xx__ReadResponse function is for reading a response from the PXI 6161 controller.

Format:

```
S32 Pxi61xx__ReadResponse(U8  Device,
                          U8 *pData,
                         U32 *DataLength);
```

Parameters:

Device

Index of the PXI 6161 board, beginning left with 1

(but counting including all other PXI 61xx/ PCI 61xx boards in the same system)

Pointer, for example pData, to the Reading data area,

consisting of Response header and Response bytes
(currently max. 1024 bytes per response)

DataLength

Value of the parameter before function call:
Size of the buffer pData is pointing to, in bytes

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

Description:

The Pxi61xx__ReadResponse function reads back the oldest response written by the PXI 6161 controller to the response area.

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

On calling up, the Pxi61xx__ReadResponse function continues to provide data until this list contains no more entries.

3.3.1.5 Read Response Block

The Pxi61xx__ReadResponseBlock function is for reading all available responses from the PXI 6161 controller.

Format:

```
S32 Pxi61xx__ReadResponseBlock(U8  Device,
                               U8 *pData,
                              U32 *DataLength,
                              U32 *BlockCounter);
```

Parameters:

Device

Index of the PXI 6161 board, beginning left with 1,

(but counting including all other PXI 61xx/ PCI 61xx boards in the same system)

Pointer, for example pData, to the Reading data area,

consisting of Response header and Response bytes
(currently max. 1024 Bytes per response)

DataLength

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

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

Pointer, for example BlockCounter

Number of contained individual responses

Description:

The Pxi61xx__ReadResponseBlock function reads back all responses written by the PXI 6161 controller to the response area.

3.3.2 VISA Device Driver

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

- Init
- Done
- System Info
- Transceiver Info
- Write Instruction
- Read Response

3.3.2.1 Init

The PXI61xx_Init function is for opening VISA sessions for all of the system's PXI/ PCI 61xx boards including initialization.

Format:

```
ViStatus PXI61xx_Init(ViUInt32 *CardCount);
```

Parameter:

CardCount

Number of the system's PXI/ PCI 61xx boards recognized by the VISA driver.

Description:

The PXI61xx_Init function searches for all PXI/ PCI 61xx 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.

3.3.2.2 Done

The PXI61xx_Done function closes all VISA sessions for the system's PXI/ PCI 61xx boards.

Format:

```
ViStatus PXI61xx_Done(void);
```

Parameter:

None

Description:

The PXI61xx_Done function closes all VISA sessions of the system's PXI/ PCI 61xx boards.

No further access to the boards is possible, then.

3.3.2.3 System Info

The PXI61xx_SystemInfo function provides general driver and board information.

Format:

```
ViStatus PXI61xx_SystemInfo(t_System_Info *DriverData,
                            ViUInt32  LengthInByte,
                              ViChar *DeviceName);
```

Parameters:

Pointer, for example DriverData, to a data structure
(For the structure, see the *PXI61xx_API.h* file on the supplied CD)

LengthInByte
Size of the buffer DriverData is pointing to, in bytes

DeviceName
`Array[K_DEV_MAX][K_RES_NAME_LENGTH]`
(see *PXI61xx_API.h*)

Description:

The PXI61xx_SystemInfo function provides information regarding the driver and the system's PXI/ PCI 61xx boards.

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

3.3.2.4 Transceiver Info

The PXI61xx_TransceiverInfo function provides information regarding the plugged-in transceivers as well as their number.

Format:

```
ViStatus PXI61xx_TransceiverInfo(t_Transceiver_Properties *TransceiverProperties,
                                                    ViUInt32  LengthInByte);
```

Parameters:

Pointer, for example TransceiverProperties, to a data structure

(For the structure, see the *PXI61xx_API.h* file on the supplied CD)

LengthInByte

Size of the buffer TransceiverProperties is pointing to, in bytes

Description:

The PXI61xx_TransceiverInfo function provides information regarding the transceiver properties.

For this, the address of a TransceiverProperties pointer has to be transferred to the function.
Within the function, the structure TransceiverProperties is pointing to is filled with the corresponding pieces of information.

3.3.2.5 Write Instruction

The PXI61xx_WriteInstruction function is for writing data to the PXI 6161 controller.

Format:

```
ViStatus PXI61xx_WriteInstruction(ViUInt8 *pData,
                                  ViUInt16  DataLength);
```

Parameters:

Pointer, for example pData, to the Writing data area,

consisting of Command header and Command bytes (currently max. 1024 bytes per command)

DataLength

Size if the memory area pData is pointing to, in bytes

Description:

The PXI61xx_WriteInstruction function allows writing of data to the PXI 6161 controller.

In the header of the structure pData is pointing to, there is the information regarding the PXI 6161 board to be activated by this function.
Therefore this parameter is not to be given separately.

3.3.2.6 Read Response

The PXI61xx_ReadResponse function is for reading data from the PXI 6161 controller.

Format:

```
ViStatus PXI61xx_ReadResponse(ViUInt8  Device,
                              ViUInt8 *pData,
                             ViUInt32 *DataLength);
```

Parameters:

Device

Index of the PXI 6161 board, beginning left with 1,
(but counting including all other PXI 61xx/ PCI 61xx boards in the same system)

Pointer, for example pData, to the Reading data area,
consisting of Response header and Response bytes
(currently max. 1024 bytes per response)

DataLength

Value of the parameter before function call:
Size of the buffer pData is pointing to, in bytes

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

Description:

The PXI61xx_ReadResponse function allows reading of data provided by the PXI 6161 controller (see also the ReadResponse function in the Windows Device Driver section).

3.4 Programming with LabVIEW

3.4.1 LabVIEW via the G-API

The supplied CD contains VIs for activating PXI 6161 boards under LabVIEW.

These LabVIEW VIs use the functions of the GOEPEL G-API.

3.4.2 LLB using the Windows Device Driver

The supplied CD contains VIs for activating PXI 6161 boards under LabVIEW.

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

3.4.3 LLB using the VISA Device Driver

The supplied CD contains VIs for activating PXI 6161 boards under LabVIEW.

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

3.5 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.