Produktbeschreibung

PXI / PCI 3051

CAN Interfaces
Nutzerhandbuch
Version 2.0

GÖPEL electronic

Get the total Coverage!

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

gedruckt: 17.06.2010

Stand: Juni 2010

1 Installation der Boards

1.1 Hardware Installation

Stellen Sie bitte unbedingt sicher, dass alle Installationsarbeiten im **ausgeschalteten** Zustand Ihres Systems erfolgen!

Das PCI™-, CompactPCI™- oder PXI™-System wird entsprechend seinen Gegebenheiten geöffnet. Wählen Sie einen freien Steckplatz in Ihrem System aus.
Beim ausgewählten Steckplatz entfernen Sie das vorhandene Slotblech. Dazu müssen die beiden Befestigungsschrauben gelöst werden.

(Wenn es notwendig ist, Transceivermodule zu tauschen, sind die allgemeinen Regeln zur Vermeidung von elektrostatischen Aufladungen zu beachten. Die Module dürfen nie unter Spannung gezogen oder gesteckt werden! Ein lagerichtiges Stecken der Module ist unbedingt zu realisieren.)

Fassen Sie das Board bei der Montage nur an den Rändern an. Berühren Sie niemals die Oberfläche, da sonst akute Zerstörungsgefahr durch elektrostatische Aufladung besteht.

Das Board ist vorsichtig in den vorbereiteten Steckplatz einzuführen. Mit dem an der Frontplatte befindlichen Hebel wird es das letzte Stück eingeschoben.

Nach dem Kontaktieren des Boards wird dieses mit den beiden Schrauben am Frontblech befestigt. Somit ist das Board ordnungsgemäß eingebaut. Danach sind ggf. die Arbeiten am System auszuführen, die dieses wieder betriebsbereit machen.

1.2 Treiberinstallation

1.2.1 Windows Device Treiber

Durch die Plug-and-Play Fähigkeit von Windows® 2000/ XP wird für jede neu erkannte Hardwarekomponente automatisch über den Hardwareassistenten eine Treiberinstallation gestartet.
Mit der auf der beiliegenden CD enthaltenen *inf*-Datei kann der Hardwareassistent die Installation des Devicetreibers durchführen.

Ein Neustart des Systems ist nicht zwingend erforderlich.

Der zur Verfügung stehende Devicetreiber unterstützt gegenwärtig ausschließlich Windows® 2000/ XP-Systeme!

Wenn Sie eigene Software für die Boards erstellen wollen, benötigen Sie ggf. zusätzliche Dateien für die anwenderspezifische Programmierung (**.LLB*, **.H*). Diese werden nicht automatisch übernommen und müssen deshalb manuell von der mitgelieferten CD in Ihr Entwicklungsverzeichnis kopiert werden.

Die I/O-Basisadresse wird während des Bootvorgangs des Systems generiert und in den Konfigurationsbereich des Boards geschrieben. Eine manuelle Einstellung ist nicht notwendig.

Interrupts und DMA-Kanäle werden für diese Boards nicht benötigt.

1.2.2 VISA Device Treiber

1. Schritt

Kopieren Sie den mitgelieferten Ordner *VISA_Driver* von der mitgelieferten CD auf die Festplatte.

(Empfehlung: vollständigen Ordner auf *C:* \)

2. Schritt

WindowsNT :

Anschließend öffnen Sie das Unterverzeichnis *C:* *VISA_Driver* *PXI3051* *Installation* und installieren die Datei *PXI3051_NT4.inf* (Datei markieren und mit der rechten Maustaste Popup Menü öffnen; darin den Auswahlpunkt *Installieren* ausführen).

Beachten Sie, dass während der Installation der *inf* -Datei keine installationsrelevanten Informationen angezeigt werden!

Windows98, Windows2000, WindowsXP :

Durch die Plug-and-Play Fähigkeit wird für jede neu erkannte Hardwarekomponente automatisch über den Hardwareassistenten eine Treiberinstallation gestartet. Folgen Sie den Anweisungen und geben Sie bei der Suche nach dem Treiber das Zielverzeichnis an, in dem sich die **.inf* -Dateien befinden
(nach Empfehlung: *C:* *VISA_Driver* *PXI3051* *Installation*).

Dabei ist für Windows98 die Datei *PXI3051_9x.inf* zu wählen, für Windows2000/XP die Datei *PXI3051_NT5.inf*.

LabViewRT :

Für den Einsatz der PXI/ PCI 3051-Boards unter dem RT Betriebssystem muss die Datei *P3051_RT.inf* im Verzeichnis *C:* *VISA_Driver* *PXI3051* *Installation* verwendet werden.

Kopieren Sie diese Datei in das Verzeichnis *ni-rt* *system* des embedded controllers (Empfehlung: über den NI Measurement Explorer).

Zum ggf.späteren Erstellen einer *startup.rtexe* sollte auch die Datei *cvi_lvrt.dll* in das Verzeichnis *ni-rt* *system* kopiert werden.

3. Schritt:

Nach einem Neustart des Computers ist die Installation abgeschlossen.

Nach der Treiberinstallation können Sie überprüfen, ob die Boards einwandfrei vom System eingebunden worden sind:

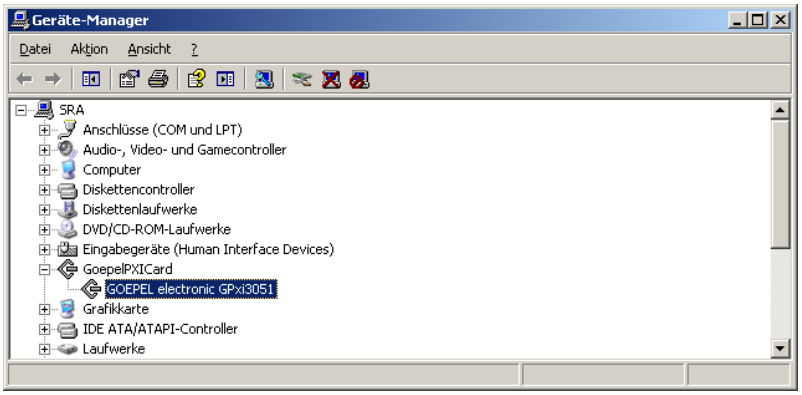

Abbildung 1-1: Windows

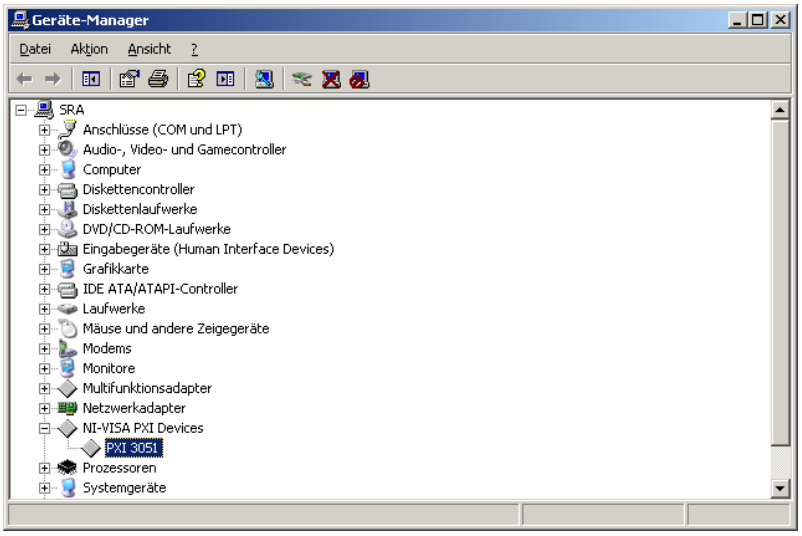

Abbildung 1-2: Visa

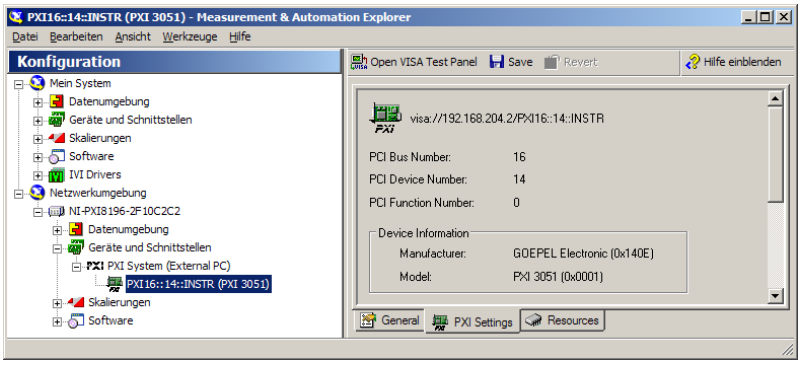

Abbildung 1-3: Visa RT

2 Hardware PXI / PCI 3051

2.1 Bestimmung

Die CAN Interface Boards PXI 3051/ PCI 3051 sind Kommunikationsboards der GÖPEL electronic GmbH.

Diese Boards mit zwei bis vier CAN Schnittstellen je nach Ausbaustufe werden in der allgemeinen Steuerungstechnik, speziell in der Automobiltechnik, verwendet.

In diesem Nutzerhandbuch ist unter Controller IMMER der einer CAN-Schnittstelle zugeordnete Microcontroller zu verstehen (unabhängig von der Bezeichnung „CAN Controller“ für das gesamte Board auf der Frontplatte).

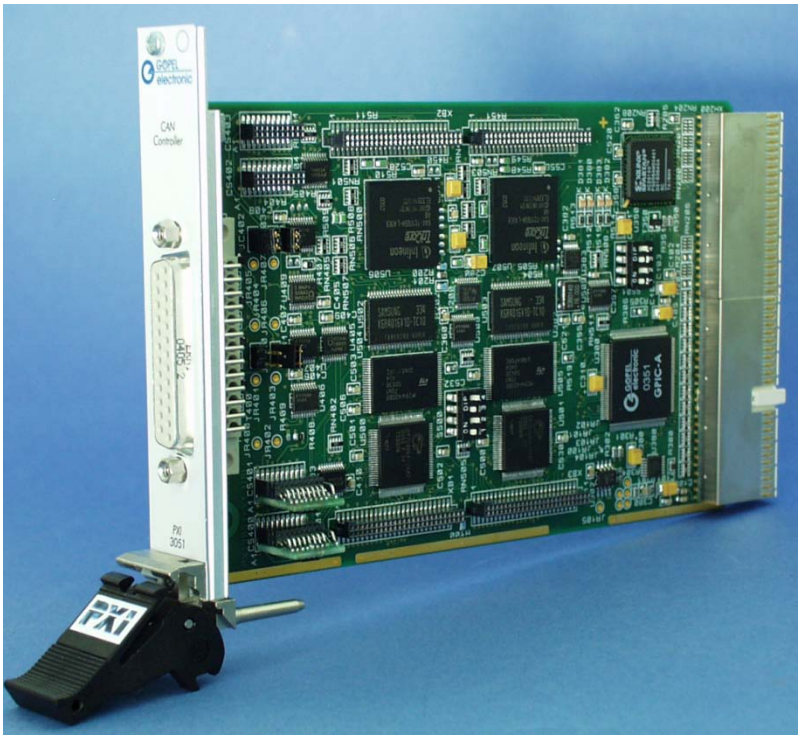

Abbildung 2-1: Basisboard PXI 3051

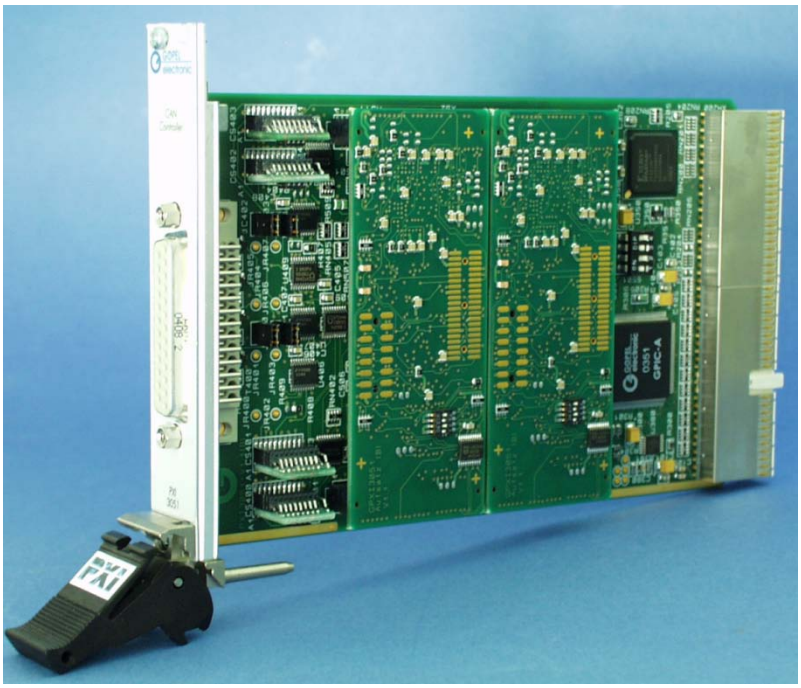

Abbildung 2-2:
PXI 3051 mit 4 x CAN

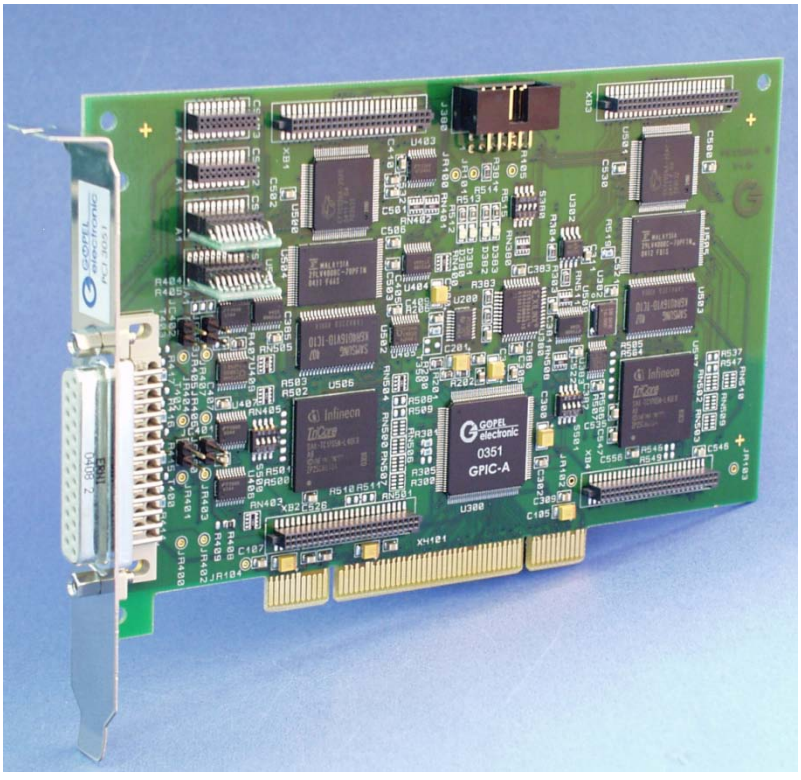

Abbildung 2-3:
Basisboard PCI 3051

2.2 Technische Daten

2.2.1 Allgemeines

Das Kommunikationsboard PXI 3051 ist ein Einsteckboard, das für den PXI™-Bus (**P**CI e**X**tensions for **I**nstrumentation) entwickelt wurde. Basis für diesen Bus ist der CompactPCI™-Bus. Es ist möglich, das Board in einem CompactPCI™- oder einem PXI™-System zu betreiben. Dieses Board kann in jeden beliebigen Steckplatz (ausgenommen Steckplatz 1) eines solchen Systems gesteckt werden. Es ist auch bei gleichzeitigem Gebrauch mehrerer Boards dieses Typs in einem Rack eindeutig identifizierbar.

Das Kommunikationsboard PCI 3051 ist ein PC-Einsteckboard für den PCI Local Bus Rev. 2.2 und kann in jedem beliebigen PCI-Steckplatz (32Bit, 33MHz, 3,3V) betrieben werden.

Beide Boards haben keine Jumper zur Hardwareerkennung und werden automatisch in das jeweilige System eingebunden.

2.2.2 Abmessungen

Die Abmessungen beider Boards entsprechen Standard-Abmessungen des zugehörigen Bussystems:

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

2.2.3 PXI 3051/ PCI 3051 Kennwerte

Symbol	Kennwert	Min.	Typ.	Max.	Einheit	Bemerkung
U_{BAT}	Batteriespannung		12	27/ 50	V	abh. v. Transceivertyp
	Übertragungsrate			1	MBaud	
R_{bus}	Abschlusswiderstand 1		120		Ohm	Jumper J1..J4 gesteckt
R_{bus}	Abschlusswiderstände 2		5,1		kOhm	auf Transceiver-Modul
	Externer Triggereingang	3,3		50	V	
	Externer Triggerausgang			U_{BAT}	V	Open Collector Ausgang über npn-Transistor: max. 50mA/ 200mW

Um am externen Triggerausgang (open Collector) einen Spannungshub zu erzeugen, muss ein externer Pullup-Widerstand über eine Spannungsquelle an diesen Ausgang geschaltet werden (z. B. 10k**Ω** über U_{Bat}).

2.3 Aufbau

2.3.1 Allgemeines

Beide Boards verfügen in der Basisversion über zwei CAN Schnittstellen der Version 2.0b und können über Aufsatzboards und weitere Transceiver auf insgesamt vier CAN Schnittstellen erweitert werden.

Abbildung 2-4 zeigt den schematischen Aufbau der Boards in einem Blockschaltbild.

Bei den PXI/ PCI 3051-Boards dient ein ASIC als Interface zum PCI- oder cPCI-Bus. Dieser beinhaltet alle notwendigen Funktionsblöcke, die für eine Kommunikation mit dem Rechner-Bus notwendig sind.

Das Board PCI 3051 besitzt KEIN PXI-Interface. Um dennoch Triggersignale mit anderen PCI-Boards von GÖPEL electronic auszutauschen, befindet sich ein zusätzlicher Steckverbinder mit zwei jeweils als Input oder Output konfigurierbaren Leitungen auf diesem Board (Trig.con in Abbildung 2-6).

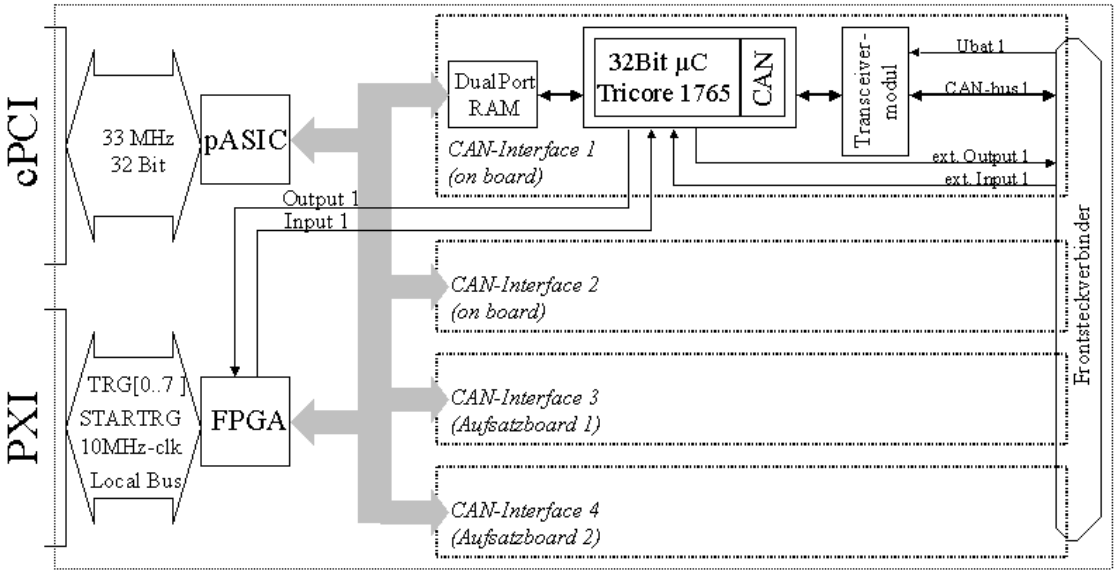

Abbildung 2-4: Blockdiagramm eines Kommunikationsboards PXI/ PCI 3051

2.3.2 Adressierung

PXI 3051: PXI-Racks besitzen eine eigene geographische Slotadressierung der Backplane. Die Nummerierung beginnt mit 1 und ist auf der Gehäusefrontseite sichtbar. Steckplatz 1 ist immer mit einem embedded Controller oder einer MXI-Karte zu bestücken.

Ein PXI 3051 Board kann die geographische Slotadresse auslesen. Hierzu muss der XILINX mit dem zugehörigen FPGA File geladen sein (siehe Funktionen XilinxDownload für unterschiedliche Treiber im Abschnitt Programmieren über DLL-Funktionen).

PCI 3051: PCI-Racks besitzen keine geographische Slotadressierung. Um dennoch mehrere PCI 3051 auf den Steckplätzen eindeutig identifizieren zu können, verfügt das Board über ein separates Adressjumper-Feld (Addr.jumper in Abbildung 2-6). Damit ist es möglich, bis zu 16 Adressvarianten zu wählen. Der mit diesem Jumperfeld gesetzte Binärwert (0..15) kann mit der Software ausgelesen werden.

2.3.3 Triggerverhalten

Jede CAN Schnittstelle besitzt 2 x 2 zusätzliche Input- bzw. Outputleitungen.

Je ein Input und Output pro CAN Schnittstelle befindet sich auf dem Frontsteckverbinder. Der jeweils zweite In- und Output kann über die entsprechende Treiberkonfiguration mit den PXI-Signalen Startrigger und Trigger[0..7] bzw. den beiden zusätzlichen I/O-Leitungen des PCI-Boards verschaltet werden.

Die Firmware der Controller kann so konfiguriert werden, dass die Schnittstelle entweder zu den externen Triggersignalen oder zu den PXI-Triggersignalen aktiviert ist.

Die folgenden Funktionen sind möglich:

- ENABLE-Funktion:
  ein von außen anliegendes Inputsignal aktiviert/ deaktiviert die Möglichkeit des Sendens von Botschaften einer CAN Schnittstelle
- TRIGGER_IN-Funktion:
  ein von außen anliegendes Triggersignal löst das Senden von zuvor definierten vorbereiteten CAN Botschaften aus
- TRIGGER_OUT-Funktion:
  es wird ein Ausgangssignal beim Senden bzw. Empfang einer bestimmten Botschaft erzeugt.

2.3.4 Kommunikationsschnittstellen

Maximal 4 x CAN Schnittstellen Version 2.0b:

Für die uneingeschränkte Funktion einer CAN Schnittstelle in einem Netzwerk ist der verwendete Transceiver entscheidend. Häufig funktionieren CAN Netzwerke nur, wenn alle Teilnehmer kompatible Transceiver im Netz haben.

Damit die Nutzer eines PXI/ PCI 3051-Boards keinen Einschränkungen unterliegen, sind die Transceiver als steckbare Module ausgeführt. Dabei stehen verschiedene Varianten (Highspeed, Lowspeed, Single-Wire u.a.) zur Auswahl, die einfach auszutauschen sind.

Neben dem Transceiver ist der Busabschlusswiderstand für die einwandfreie Funktion des CAN-Netzwerkes wichtig.
Werden Highspeed Transceiver verwendet, ist i. Allg. ein 120 Ohm Widerstand für jede CAN-Schnittstelle auf dem Board bestückt. Diese Widerstände können durch Ziehen der Jumper J1..J4 deaktiviert und durch bedrahtete Widerstände mit dem gewünschten Wert an den Positionen JP1..JP4 ersetzt werden (siehe Abbildung 2-5 und Abbildung 2-6).

Bei Verwendung von Lowspeed Transceivern befinden sich zwei Abschlusswiderstände von 5,1 kOhm für RTH und RTL pro CAN Schnittstelle auf dem Transceiver-Modul. In diesem Fall darf KEIN bedrahteter Widerstand bestückt werden, und der Jumper muss geöffnet sein.

Für die folgenden Transceivertypen ist der Anschluss der Batteriespannung an die Pins 15, 18, 21 bzw. 24 des Steckverbinders XS1 (V_Bat1..V_Bat4, siehe Belegung Frontsteckverbinder) für die jeweilige CAN-Schnittstelle notwendig:

- TJA104A
- TJA1054
- 82C52
- B10011S

2.3.5 Bestückung

Abbildung 2-5 und Abbildung 2-6 zeigen schematisch die Bestückungsseite der Boards. Dabei ist die Lage der optionalen Aufsatzboards sowie der Transceivermodule für jede Schnittstelle zu erkennen.
Die Positionen der Jumper für die Aktivierung/ Deaktivierung der Bus-Abschlusswiderstände bzw. der optional zu bestückenden bedrahteten Widerstände sind ebenfalls ersichtlich. Ein gesteckter Jumper bedeutet, dass der Bus-Abschlusswiderstand von 120 Ohm aktiv ist.

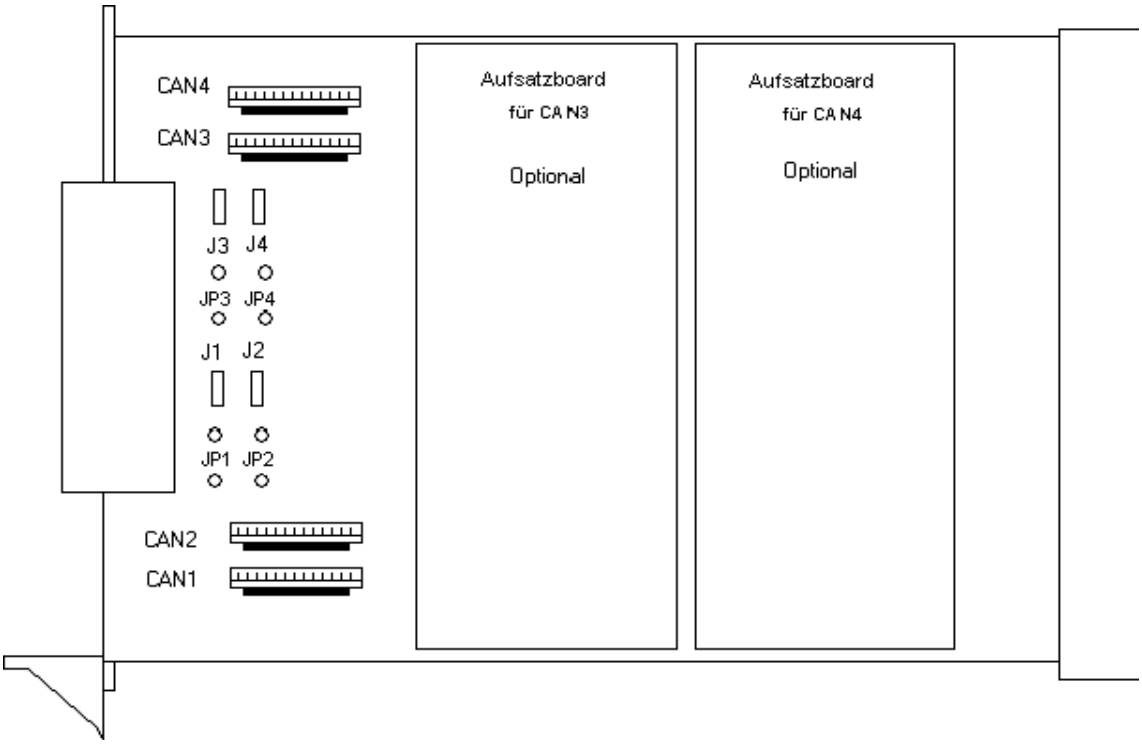

Abbildung 2-5: Schematischer Bestückungsplan Kommunikationsboard PXI 3051

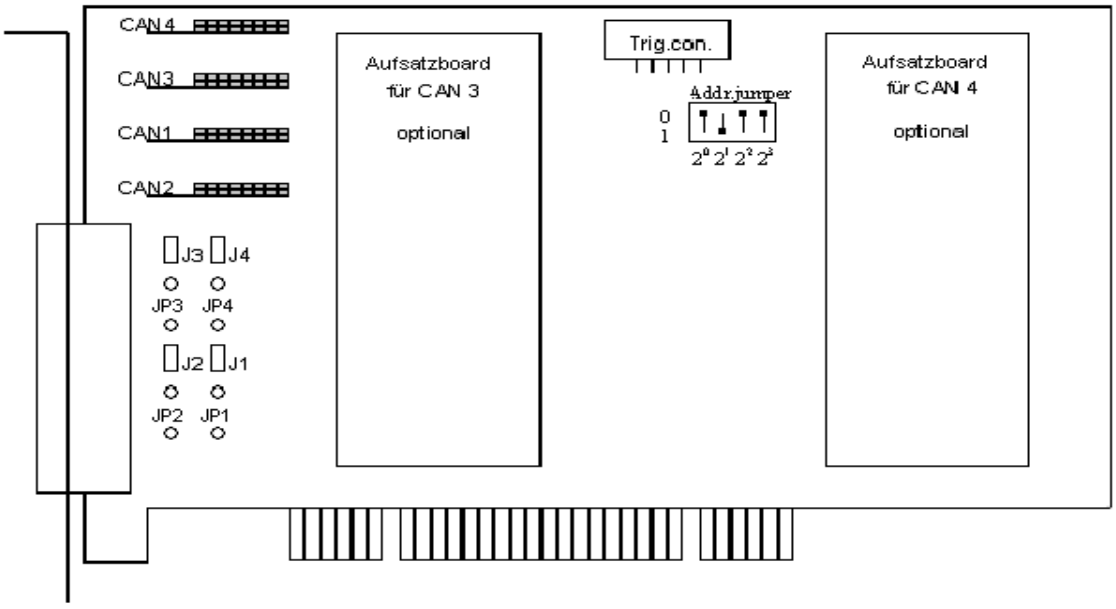

Abbildung 2-6: Schematischer Bestückungsplan Kommunikationsboard PCI 3051

Die Konfigurationselemente aus Abbildung 2-5 und Abbildung 2-6 werden in der folgenden Tabelle erläutert:

CAN1	Transceivermodul für CAN1
CAN2	Transceivermodul für CAN2
CAN3	Transceivermodul für CAN3
CAN4	Transceivermodul für CAN4
J1	Jumper zum Aktivieren des **120Ω** Busabschlusswiderstandes (onboard) für CAN1
J2	Jumper zum Aktivieren des **120Ω** Busabschlusswiderstandes (onboard) für CAN2
J3	Jumper zum Aktivieren des **120Ω** Busabschlusswiderstandes (onboard) für CAN3
J4	Jumper zum Aktivieren des **120Ω** Busabschlusswiderstandes (onboard) für CAN4
JP1	Position für optionalen bedrahteten Abschlusswiderstand – CAN1
JP2	Position für optionalen bedrahteten Abschlusswiderstand – CAN2
JP3	Position für optionalen bedrahteten Abschlusswiderstand – CAN3
JP4	Position für optionalen bedrahteten Abschlusswiderstand – CAN4
Addr.jumper	Das Jumperfeld auf der PCI 3051 dient der eindeutigen Adressierung des Boards in einem System mit mehreren PCI 3051 (analog „geografische Adressierung“ der PXI-Spezifikation). Dazu kann ein entsprechender Binärwert (0..15) über das Jumperfeld eingestellt werden, der über die mitgelieferte Software ausgelesen wird.
Trig.con	Steckverbinder auf der PCI 3051 zum Austausch von Triggersignalen mit anderen GÖPEL electronic PCI-Boards

2.3.6 Belegung Front-steckverbinder

Typ: DSub 25-polig Buchse

Die CAN Schnittstellen stehen über diesen Steckverbinder an der Frontseite der Kommunikationsboards zur Verfügung.

Die Belegung ist bei beiden Boards identisch und in der folgenden Tabelle dargestellt:

lfd. Nr.	Anschluss XS1	Signalname	Bemerkung
1	14	CAN1_High	CAN-Bus-Leitung High
2	2	CAN1_Low	CAN-Bus-Leitung Low
3	15	V_Bat1	Bezugspotenzial Plus Transceiver
4	1	GND	Massepotenzial Transceiver
5	17	CAN2_High	CAN-Bus-Leitung High
6	5	CAN2_Low	CAN-Bus-Leitung Low
7	18	V_Bat2	Bezugspotenzial Plus Transceiver
8	4	GND	Massepotenzial Transceiver
9	20	CAN3_High	CAN-Bus-Leitung High
10	8	CAN3_Low	CAN-Bus-Leitung Low
11	21	V_Bat3	Bezugspotenzial Plus Transceiver
12	7	GND	Massepotenzial Transceiver
13	23	CAN4_High	CAN-Bus-Leitung High
14	11	CAN4_Low	CAN-Bus-Leitung Low
15	24	V_Bat4	Bezugspotenzial Plus Transceiver
16	10	GND	Massepotenzial Transceiver
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	Massepotenzial

2.4 Lieferhinweise

PXI/ PCI 3051-Boards werden in folgenden Varianten geliefert:

- 2x CAN Schnittstelle
- 3x CAN Schnittstelle
- 4x CAN Schnittstelle

Zur jeweiligen CAN Schnittstelle muss auch der Typ des zugehörigen Transceivers festgelegt werden.

Für jede CAN-Schnittstelle sind außerdem die erforderlichen Funktionalitäten anzugeben.

3 Ansteuersoftware

Zur Einbindung der PXI 3051/ PCI 3051-Hardware in eigene Applikationen existieren drei Möglichkeiten:

- Programmieren über G-API
- Programmieren über DLL-Funktionen
- Programmieren mit LabVIEW

3.1 Programmieren über G-API

Das bevorzugte User Interface für diese GÖPEL Hardware ist die G-API (GÖPEL-API).

Sie finden alle benötigten Informationen im Ordner *G-API* der mitgelieferten CD.

3.2 Programmieren über DLL-Funktionen

Die Programmierung über DLL-Funktionen ist weiterhin für bestehende Projekte möglich, bei denen noch nicht mit der GÖPEL G-API gearbeitet werden kann.

Die Dokumentation GÖPEL Firmware senden wir Ihnen auf Anforderung gern zu. Bitte setzten Sie sich bei Bedarf mit unserem Vertrieb in Verbindung.

Die Begriffe GPxi3051 und PXI3051 in der folgenden Funktionsbeschreibung stehen für PXI 3051/ PCI 3051.

Informationen zu den Strukturen, Datentypen und Error-Codes enthalten die Header – die entsprechenden Dateien finden Sie auf der mitgelieferten CD.

In diesem Nutzerhandbuch ist unter Controller IMMER der einer CAN-Schnittstelle zugeordnete Microcontroller zu verstehen (unabhängig von der Bezeichnung „CAN Controller“ für das gesamte Board auf der Frontplatte).

3.2.1 Windows Device Treiber

Die für die Programmierung unter Verwendung des Windows Device Treibers nutzbaren DLL-Funktionen sind in den folgenden Abschnitten beschrieben:

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

3.2.1.1 *DriverInfo*

Die Funktion GPxi3051_GetDriverInfo dient zur Status-Abfrage des Hardware-Treibers.

Format:

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

Parameter:

Zeiger, z.B. pDriverInfo
auf eine Datenstruktur
Zur Struktur siehe das File *GPxi3051.h* auf der mitgelieferten CD

Beschreibung:

Die Funktion GPxi3051_GetDriverInfo gibt Informationen über den Status des Hardware-Treibers zurück.

Dazu muss der Funktion die Adresse eines Zeigers pDriverInfo übergeben werden. Innerhalb der Funktion wird die Struktur, auf die pDriverInfo zeigt, mit verschiedenen Informationen gefüllt.

3.2.1.2 DLL Version

Die Funktion GPxi3051_DLL_Version dient zur Abfrage der Versionsnummer der DLL.

Format:

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

Parameter

Zeiger, z.B. pVersion
auf die Versionsnummer

Beschreibung:

Die Funktion GPxi3051_DLL_Version gibt die Versionsnummer der *GPxi3051w.dll* als Integer-Wert zurück.

Beispiel:

Die Versionsnummer **`1.23`** wird als Wert **`123`** zurückgegeben, Version **`1.60`** als Wert **`160`**.

3.2.1.3 XILINX – Download

Die Funktion GPxi3051_XilinxDownload dient zum Laden eines FPGA-Files in den XILINX.

Format:

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

Parameter:

card

Index des PXI/ PCI 3051-Boards, links beginnend mit 1

Zeiger, z.B. pFileName
auf den Pfad des zu ladenden FPGA-Files

Beschreibung:

Die Funktion GPxi3051_XilinxDownload dient zum Laden eines FPGA-Files (Extension **.cfd*), das unter anderem das Auslesen der geografischen Slotadresse im PXI-Rack ermöglicht.

Die geladenen Daten sind flüchtig. Deshalb muss die Funktion nach Power Off erneut ausgeführt werden.

Nach XilinxDownload ist eine Wartezeit von ca. 500 ms erforderlich, da die Controller ein Power-On-Reset durchlaufen.

Anschließend ist der Firmware-Befehl 0x10 Software Reset für alle Controller auszuführen, um vom Bootloader-Modus in den Normal-Modus zu gelangen.

3.2.1.4 XILINX – Write Data

Die Funktion GPxi3051_XilinxWriteData ermöglicht das Konfigurieren und Ausführen von Funktionen, die der XILINX bereitstellt.

Format:

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

Parameter:

Zeiger, z.B. data
auf den Bereich für Schreibdaten

length

Größe des Speicherbereiches, auf den data zeigt, in Bytes
(z. Zt. max. 128 Byte pro Befehl)

Beschreibung:

Bevor die Funktionalität des XILINX genutzt werden kann, muss das zugehörige FPGA-File mit GPxi3051_XilinxDownload geladen worden sein (siehe XILINX – Download).

Das Datenformat besteht aus vier Bytes einschließlich Befehl.
Falls erforderlich, können Parameter-Bytes folgen.

Datenformat:
1. Byte: 0x48 **(**StartByte**)**
2. Byte: card (Index des PXI/ PCI 3051-Boards, links beginnend mit 1)
3. Byte: 0x00 **(Reserviertes Byte)**
4. Byte: XILINX Befehl

z. Zt. unterstützter XILINX Befehl:

0x10 PowerOnReset für das komplette Board

3.2.1.5 DPRAM – Write Instruction

Die Funktion GPxi3051_DpramWriteInstruction dient zum Senden eines Befehls zum ausgewählten Controller.

Format:

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

Parameter:

Zeiger, z.B. data
auf den Bereich für Schreibdaten,
bestehend aus Befehlskopf und Befehlsbytes
(z. Zt. max. 1024 Byte pro Befehl)

length

Größe des Speicherbereiches, auf den data zeigt, in Bytes

Beschreibung:

Die Funktion GPxi3051_DpramWriteInstruction sendet einen Befehl zum ausgewählten Controller.

Im Header der Struktur, auf die data zeigt, befinden sich die Informationen zum anzusprechenden PXI/ PCI 3051-Board und zum anzusprechenden Controller.
Deshalb sind die entsprechenden Parameter nicht separat anzugeben.

3.2.1.6 DPRAM – Read Response

Die Funktion GPxi3051_DpramReadResponse dient zum Lesen einer Antwort vom ausgewählten Controller.

Format:

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

Parameter:

card

Index des PXI/ PCI 3051-Boards, links beginnend mit 1

port

Nummer des Controllers (1..4)

Zeiger, z.B. data
auf den Bereich für Lesedaten,
bestehend aus Antwortkopf und Antwortbytes
(z. Zt. max. 1024 Byte pro Antwort)

length

Parameterwert vor Funktionsaufruf:
Größe des Puffers, auf den data zeigt, in Bytes

Parameterwert nach Funktionsaufruf:
Tatsächlich gelesene Byteanzahl

Beschreibung:

Die Funktion GPxi3051_DpramReadResponse liest die älteste vom Controller (1..4) im Response-Bereich des DPRAM geschriebene Antwort zurück.

Werden mehrere Antworten vom jeweiligen Controller bereitgestellt, ohne sie zu senden, gehen diese nicht verloren, sondern werden in einer Art Liste abgelegt.

Aufrufe von GPxi3051_DpramReadResponse liefern dann solange Werte, bis diese Liste keine Einträge mehr enthält.

3.2.1.7 DPRAM – Read Monitor

Die Funktion GPxi3051_DpramReadMonitor dient zum Lesen der Monitordaten des ausgewählten Controllers.

Format:

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

Parameter:

card

Index des PXI/ PCI 3051-Boards, links beginnend mit 1

port

Nummer des Controllers (1..4)

Zeiger, z.B. data
auf den Bereich für Lesedaten (max. 20kByte)

length

Parameterwert vor Funktionsaufruf:
Größe des Puffers, auf den data zeigt, in Bytes

Parameterwert nach Funktionsaufruf:
Anzahl der tatsächlich gelesenen Monitoreinträge

Beschreibung:

Die Funktion GPxi3051_DpramReadMonitor liest die im Monitorbereich des DPRAM befindlichen Daten.

Dabei handelt es sich ausschließlich um die Daten, die im Monitormode Pufferempfang vom ausgewählten Controller bereitgestellt werden. Das heißt, dass der normale Response-Bereich vom Daten-Bereich des Monitors (Pufferempfang) getrennt ist.

Pro Monitoreintrag werden 20 Byte benötigt. Die zurückgegebene Länge length ist bereits durch diese 20 Byte geteilt, entspricht somit der Anzahl der rückgelesenen Monitoreinträge.

3.2.1.8 Reset Port

Die Funktion GPxi3051_ResetPort dient zur Auslösung eines Software-Resets für den ausgewählten Controller.

Format

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

Parameter:

card

Index des PXI/ PCI 3051-Boards, links beginnend mit 1

port

Nummer des Controllers (1..4)

Beschreibung:

Die Funktion GPxi3051_ResetPort löst ein Software-Reset für den ausgewählten Controller aus.

Dieser Auslösemechanismus erfolgt über einen separaten Interruptkanal und nicht über den Befehlsinterpreter der Software (Firmware-Befehl 0x10 Software Reset).

3.2.2 VISA Device Treiber

Die für die Programmierung unter Verwendung des VISA Device Treibers nutzbaren DLL-Funktionen sind in den folgenden Abschnitten beschrieben:

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

3.2.2.1 Init

Die Funktion PXI3051_Init dient zur Eröffnung von VISA Sessions für alle im System befindlichen PXI/ PCI 3051-Boards und deren Initialisierung.

Format:

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

Parameter:

CardCount

Anzahl der vom VISA Treiber erkannten PXI/ PCI 3051-Boards im System

Beschreibung:

Die Funktion PXI3051_Init sucht alle im System befindlichen PXI/ PCI 3051-Boards und eröffnet die erforderlichen Sessions.
Außerdem werden Board-interne Initialisierungen durchgeführt.

Deshalb muss diese Funktion als erster Schritt ausgeführt werden.

3.2.2.2 Done

Die Funktion PXI3051_Done schließt alle VISA Sessions für im System befindliche PXI/ PCI 3051-Boards.

Format:

```
ViStatus PXI3051_Done(void)
```

Parameter:

keine

Beschreibung:

Die Funktion PXI3051_Done schließt alle VISA Sessions für im System befindliche PXI/ PCI 3051-Boards.

Damit ist kein weiterer Boardzugriff möglich.

3.2.2.3 *Driver Info*

Die Funktion PXI3051_DriverInfo liefert allgemeine Treiber- und Boardinformationen.

Format:

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

Parameter:

Zeiger, z.B. DriverData
auf eine Datenstruktur
Zur Struktur siehe das File *PXI3051_API.h* der mitgelieferten CD

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

Beschreibung:

Die Funktion PXI3051_DriverInfo stellt verschiedene Informationen zum Treiber und zu den im System befindlichen PXI/ PCI 3051-Boards zur Verfügung.

Der DeviceName gibt die von VISA erfassten Ressourcennamen an. Diese Informationen korrellieren mit der Anzeige im NI MAX.

3.2.2.4 XILINX – Download

Die Funktion PXI3051_XilinxDownload dient zum Laden eines FPGA Files in den XILINX.

Format:

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

Parameter:

Card

Index des PXI/ PCI 3051-Boards, links beginnend mit 1

Zeiger, z.B. FileName
auf den Pfad des zu ladenden FPGA Files

Beschreibung:

Die Funktion GPxi3051_XilinxDownload dient zum Laden eines FPGA-Files (Extension **.cfd*), das unter anderem das Auslesen der geografischen Slotadresse im PXI-Rack ermöglicht.

Die geladenen Daten sind flüchtig. Deshalb muss die Funktion nach Power Off erneut ausgeführt werden.

Nach XilinxDownload ist eine Wartezeit von ca. 500 ms erforderlich, da alle Controller ein Power-On-Reset durchlaufen.

Anschließend ist der Firmware-Befehl 0x10 Software Reset auf allen Controllern auszuführen, um vom Bootloader-Modus in den Normal-Modus zu gelangen.

3.2.2.5 XILINX – Write Data

Die Funktion PXI3051_XilinxWriteData ermöglicht das Konfigurieren und Ausführen von Funktionen, die der XILINX bereitstellt.

Format:

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

Parameter:

Zeiger, z.B. WriteData
auf den Bereich für Schreibdaten (z. Zt. max. 128 Byte pro Befehl)

Beschreibung:

Bevor die Funktionalität des XILINX genutzt werden kann, muss das zugehörige FPGA-File mit PXI3051_XilinxDownload geladen worden sein (siehe XILINX Download).

Das Datenformat besteht aus vier Bytes einschließlich Befehl.
Falls erforderlich, können Parameter-Bytes folgen.

Datenformat:
1. Byte: 0x48 (StartByte)
2. Byte: card (Index des PXI/ PCI 3051-Boards, links beginnend mit 1)
3. Byte: 0x00 (Reserviertes Byte)
4. Byte: XILINX Befehl

z. Zt. unterstützter XILINX Befehl:

0x10 PowerOnReset für das komplette Board

3.2.2.6 Write Data

Die Funktion PXI3051_WriteData dient zum Schreiben von Daten zum ausgewählten Controller.

Format:

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

Parameter:

Zeiger, z.B. WriteData
auf den Bereich für Schreibdaten,
bestehend aus Befehlskopf und Befehlsbytes
(z. Zt. max. 1024 Byte pro Befehl)

Length_In_Bytes

Größe des Speicherbereiches, auf den WriteData zeigt, in Bytes

Beschreibung:

Die Funktion PXI3051_WriteData ermöglicht das Schreiben von Daten zum ausgewählten Controller.

Im Header der Struktur, auf die WriteData zeigt, befinden sich die Informationen zum anzusprechenden PXI/ PCI 3051-Board und zum anzusprechenden Controller.
Deshalb sind sie nicht gesondert als Parameter anzugeben.

3.2.2.7 Read Data

Die Funktion PXI3051_ReadData dient zum Lesen von Daten vom ausgewählten Controller.

Format:

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

Parameter:

Card

Index des PXI/ PCI 3051-Boards, links beginnend mit 1

Port

Nummer des Controllers (1..4)

Zeiger, z.B. ReadData
auf den Bereich für Lesedaten,
bestehend aus Antwortkopf und Antwortbytes
(z. Zt. max. 1024 Byte pro Antwort)

Length

Parameterwert vor Funktionsaufruf:
Größe des Puffers, auf den ReadData zeigt, in Bytes

Parameterwert nach Funktionsaufruf:
Anzahl der tatsächlich gelesenen Bytes

Beschreibung:

Die Funktion PXI3051_ReadData ermöglicht das Lesen von Daten, die vom ausgewählten Controller bereitgestellt wurden
(siehe auch Funktion PXI3051_DpramReadResponse im Abschnitt Windows Device Treiber).

3.2.2.8 Read Monitor

Die Funktion PXI3051_ReadMonitor dient zum Lesen von Monitordaten des ausgewählten Controllers.

Format:

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

Parameter:

Card

Index des PXI/ PCI 3051-Boards, links beginnend mit 1

Port

Nummer des Controllers (1..4)

Zeiger, z.B. MonitorData
auf den Bereich für Lesedaten (max. 20kByte)

Length

Parameterwert vor Funktionsaufruf:
Größe des Puffers, auf den MonitorData zeigt, in Bytes

Parameterwert nach Funktionsaufruf:
Anzahl der tatsächlich gelesenen Monitoreinträge

Beschreibung:

Die Funktion PXI3051_ReadMonitor ermöglicht das Lesen von Daten aus dem Monitorbereich des ausgewählten Controllers
(siehe auch Funktion GPxi3051_DpramReadResponse im Abschnitt Windows Device Treiber).
Dabei handelt es sich um einen separaten Lesebereich des Boards.

3.2.2.9 Reset Port

Die Funktion PXI3051_ResetPort dient zur Auslösung eines Software-Resets für den ausgewählten Controller.

Format

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

Parameter:

Card

Index des PXI/ PCI 3051-Boards, links beginnend mit 1

Port

Nummer des Controllers (1..4)

Beschreibung:

Die Funktion PXI3051_ResetPort löst ein Software-Reset für den ausgewählten Controller aus.

Dieser Auslösemechanismus erfolgt über einen separaten Interruptkanal und nicht über den Befehlsinterpreter der Software (Firmware-Befehl 0x10 Software Reset).

3.3 Programmieren mit LabVIEW

3.3.1 LabVIEW über G-API

Auf der mitgelieferten CD befindet sich eine VI-Sammlung, mit deren Hilfe PXI/ PCI 3051-Boards unter LabVIEW angesprochen werden können.

Dabei nutzen die LabVIEW VIs die Funktionen der GÖPEL G-API.

3.3.2 LLB unter Verwendung des Windows Device Treibers

Auf der mitgelieferten CD befindet sich eine VI-Sammlung, mit deren Hilfe PXI/ PCI 3051-Boards unter LabVIEW angesprochen werden können.

Dabei werden die Funktionen genutzt, die im Abschnitt Windows Device Treiber beschrieben worden sind.

3.3.3 LLB unter Verwendung des VISA Device Treibers

Auf der mitgelieferten CD befindet sich eine VI-Sammlung, mit deren Hilfe PXI/ PCI 3051-Boards unter LabVIEW angesprochen werden können.

Dabei werden die Funktionen genutzt, die im Abschnitt VISA Device Treiber beschrieben worden sind.

3.4 Weitere Göpel Software

PROGRESS, Programm Generator und myCAR der GÖPEL electronic GmbH sind komfortable Programme zur Prüfung mit GÖPEL-Hardware.

Weitere Informationen zur Nutzung dieser Programme finden Sie in den entsprechenden Softwarebeschreibungen.