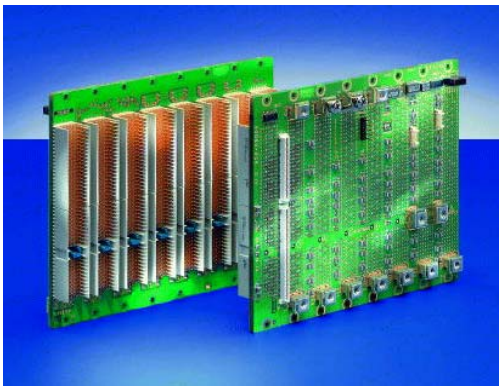
User Manual
PXI Backplanes



What is PXI?

PCI e**X**tensions for **I**nstrumentation (PXI) delivers a rugged, PC-based, and high-performance measurement and automation system. With PXI you benefit from the low cost, performance, and flexibility of the latest PC technology and the benefits of an open industry standard. PXI combines standard PC technology from the CompactPCI specification with integrated timing and triggering to deliver a rugged platform with up to a 10x performance improvement over older architectures. PXI has become the industry standard for measurement and automation applications.

PXI has seen rapid adoption over the last six years and is used in all market segments where measurement, control, or automation is required. From advanced research, military and aerospace, consumer electronics, and communication, to process control and industrial automation PXI is being selected as the platform of choice for thousands of applications.

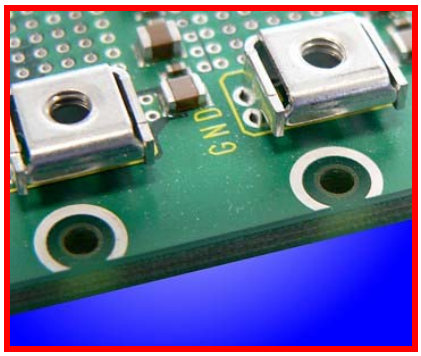
Schroff PXI Backplanes

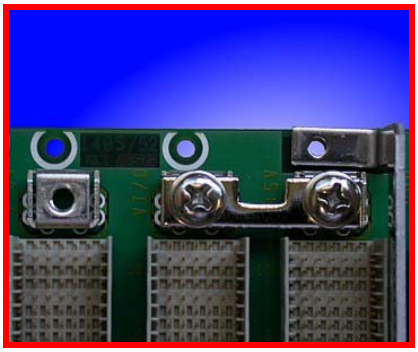
Schroff PXI backplanes are fully compliant to the latest PICMG specifications.

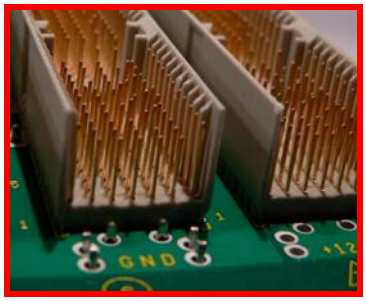
PICMG 2.0 R3.0	cPCI Core Specification
PICMG 2.1	cPCI Hot Swap Specification
PICMG 2.6	Bridging Specification
PICMG 2.8	Pin Registration for PXI
PICMG 2.9	System Management Bus Specification
PICMG 2.10	Keying Specification
PXI Spec	PXI Specification Rev. 2.0

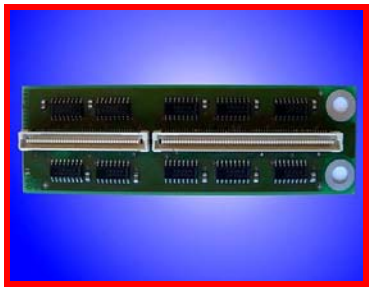
Schroff PXI backplanes are specially designed to achieve exellent power distribution, best signal integrity, virtually zero cross talk, and minimum clock skew. The SMD components used on Schroff PXI backplanes lead to a much lower failure rate than conventional components.

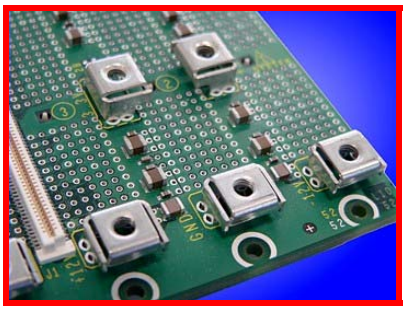
Schroff uses ceramic capacitors on the PXI backplanes to gain a better noise reduction at frequencies above 10MHz. This feature reduces the radiated and conducted noise caused by the processor and PCI clock signals. In addition, ceramic capacitors have no limitation in useful lifetime, as compared to aluminium capacitors that dry out after 5 to 10 years, and are unaware of the hazardous fire risks known from tantalum electrolytics'.











Schroff PXI Backplane Features

Isolated Assembling / Connection to ChassisGND

Schroff PXI backplanes have a specially designed pattern of mounting holes to assemble the backplane isolated or connected to ChassisGND.

For isolation between BackplaneGND and ChassisGND M2.5 screws and isolating washers should be used in at least every second connector position.

If noise reduction shall be achieved by connecting DigitalGND to ChassisGND conductive spring washers are recommended instead of isolating ones.

VI/O

Schroff PXI backplanes have a complete power plane for the VI/O voltage. The VI/O plane can be connected using a bridge on the power bugs to +3.3V or +5V.

By default, Schroff PXI backplanes have VI/O connected to +5V with blue coding keys on P1. VI/O can also be set to 3,3V with the conversion kit **21101-658** (including 8 yellow keys and a tool) and change of the VI/O bridge position on the rear side of the backplane. Schroff PXI backplanes are, on request, available with VI/O set to 3,3V.

Hot Swap

Schroff PXI backplanes fulfill the requirements for Basic Hot Swap of the Hot Swap Specification PICMG 2.1 R2.0. The signal BD_SEL# is tied to GND.

The P1 connector on Schroff cPCI backplanes has pin stagging needed for hot swap capabilities.

Termination

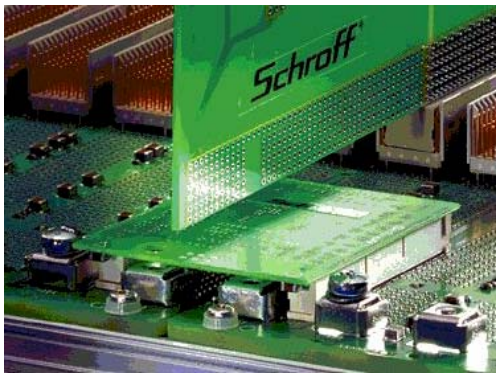
Termination on backplanes according to PICMG 2.0 R 3.0 is recommended in one case only. If, on a 8 slot backplane, strong buffers are used and only the system and the first adjacent slot are occupied and all others are empty.

Schroff has implemented a special connector on the 8 Slot PXI backplane, where a Termination Board can be assembled. For slot counts 4 to 7, this connector is used for assembling a PXI Bridge, see chapter „Mechanical and Electrical Interface".
Schroff is offering a 64-bit Termination Board, order code **23006-931**.

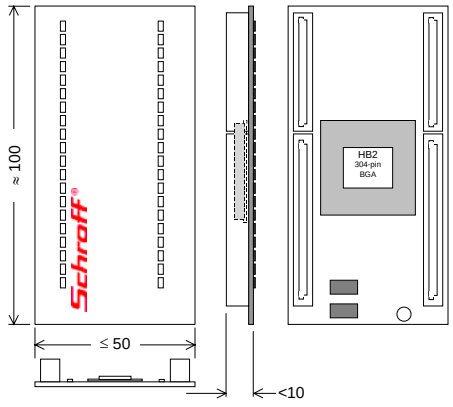
Power Bugs

Schroff PXI backplanes are populated with specially designed power bugs. The power cables can be connected to the power bugs with cable lugs fastened with M4 screws. Each power bug can handle 30 Amps.

Schroff has designed various cables and power boards to be assigned to these power bugs. They provide interfaces for many different PSU types. The power cables and power backplanes are displayed in the Schroff main catalogue and the Schroff webpage. For a detailed description, please check the cPCI manual, downloadable from the Schroff webpage.



PXI Rear Brick Bridge
64Bit, left-to-right



part#: **23006 - 924**

PXI Bridge

CPCI and PXI has been designed to accommodate up to 8 modules on a bus segment. Installing a bridge module on a bus segment consumes one of the loads on the segment but creates a new bus segment with up to 7 additional modules. The bridge module handles all communication between the bus segments.

With the rear palette bridges from Schroff no valuable front slot is wasted. Due to the very low height of 10mm, even no rear I/O slot is wasted.

Schroff offers a a 64-bit rear palette PXI bridge for system slot left backplanes.

The maximum power consumption of this bridge module is 0.75W at +5V and 2.2 W at +3.3V. GND and the power supplies (+5V, +3.3V, +/-12V) are connected from the primary PCI bus to the secondary PCI bus. V(I/O) may be +3.3V or +5V on either side of the bridge. The bridge will automatically detect the bus voltage and adjust its I/O levels accordingly.

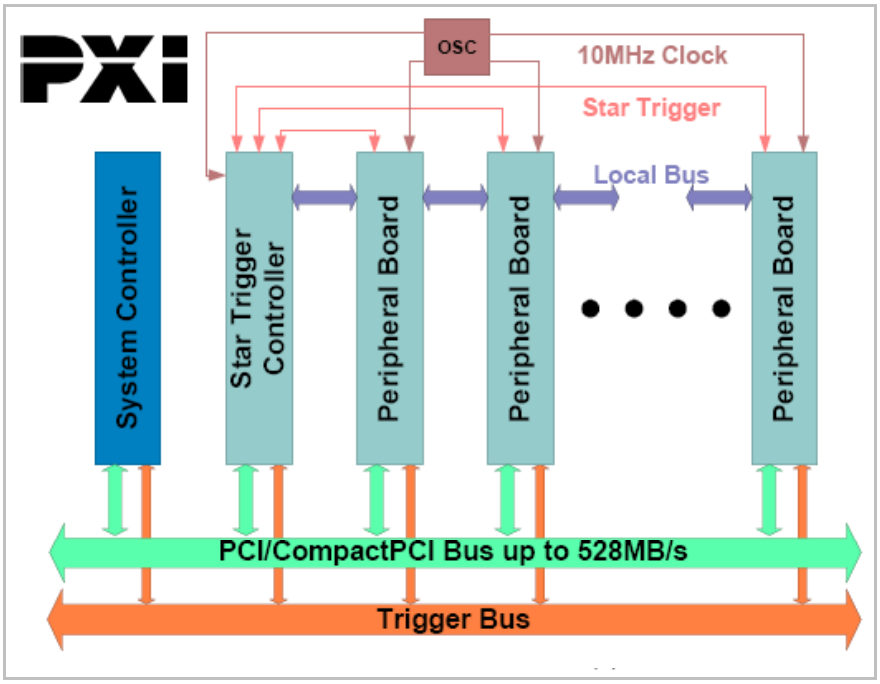
Power Supply of Backplanes: Both backplanes connected by the bridge are to be powered individually. The bridge is not to be used for bridging power. The bridge does not isolate the power rails of both backplanes. GND is connected by a sufficient number of pins in the connector to ensure signal integrity and a common GND potential on both backplanes.

VI/O There is no need to choose the VI/O voltage for the bridge. The bridge automatically takes the VI/O voltage of the primary and secondary side. Both backplanes can be set to different VI/O voltages, e.g. +5V on the primary side and 3,3V on the secondary side.

Mechanical Mounting Both backplanes should only be attached to the horizontal rails, but not fixed. The mounting screws of the backplane are not to be tightened until the bridge is plugged and fully seated!

Overview PXI Architecture and Implementation

Details of the PXI Architecture can be found in the PXI Specification, downloadable at www.pxisa.org.



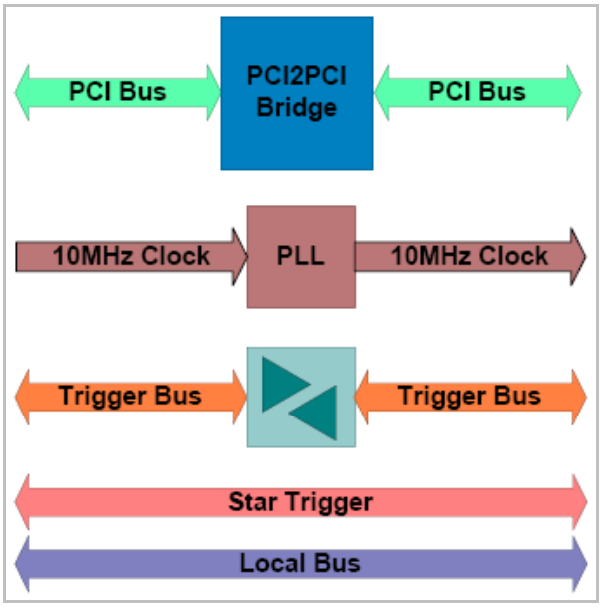
Bus Interface

PXI uses the CompactPCI/PCI Bus to transfer data between CPU and peripheral cards. The maximum throughput for a 32 Bit / 33 MHz system is 132 MB/s, the maximum throughput for a 64 Bit / 66 MHz system is 528 MB/s.

Please consider that the throughput rates mentioned above are theoretical peak values. Practical peak values are about 85% of these rates. The sustained throughput may largely differ from these peak values. It depends heavily on the kind and number of installed components and the software environment.

Due to CompactPCI compatibility, PXI and CompactPCI cards can be mixed as desired. You may use CompactPCI components for functions that do not use PXI extensions. This also includes the system controller.

The maximum throughput on the secondary and tertiary backplane segments might be lower than on the primary segment as a result of the inherent latency of these chips. Therefore cards with high throughput requirements should be installed in the primary backplane segment.



PXI Extensions

PXI Reference Clock

PXI defines a highly precise reference clock which is independent from the PCI BUS clock. The frequency is 10 MHz +/- 100 ppm. The clock signal is generated on the primary backplane and distributed to all slots except the system slot. The skew between any two slots of two backplane segments is typically 600 ps and 1.2 ns maximum.The skew between any two slots of the chassis is 900 ps typically and 1.8 ns maximum. The Star Trigger slot is the first peripheral slot to the right of the system slot.

PXI Local Bus

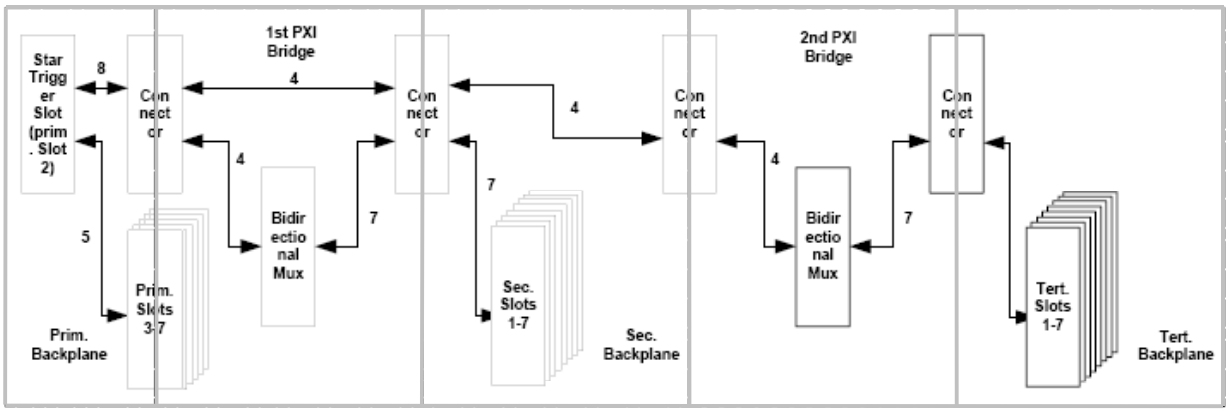
The Local Bus is routed over the backplane as a daisy chain connection. For each slot (except the Star Trigger slot) there is a definition of 13 signals connecting the slot with its left neighbour and 13 signals connecting the slot with its right neighbour. The Star Trigger slot does not have a connection to its left neighbour, the system slot, as the system slot does not access the Local Bus. The impedance of the Local Bus signal lines is 75 Ohm +/- 10%. Maximum voltage is 42 V and maximum current is 300 mA. The Local Bus can transfer digital as well as analog signals. Please consult the respective module's documentation for compliance with the other modules in use before connecting it to the Local Bus. Schroff PXI Backplanes are designed for a current carrying capacity of 400 mA. The PXI specification only demands 200 mA. If you want to make use of the higher capacity, make sure it is supported by the used peripheral boards.

PXI Trigger Bus

The Trigger Bus consists of 8 lines than run parallel between the slots of a backplane. The level of the Trigger Bus is LVTTL compatible. The PXI Trigger lines can be used to exchange clock signals of variable frequency and asynchronous or synchronous trigger signals between any units. The different trigger protocols are well described in the PXI specifications. To maintain the signal integrity of the trigger signals in Schroff PXI bridge applications, the trigger lines are buffered between the backplane segments. The buffers are bi-directional and can be independently switched for any signal. Therefore it is possible to use the trigger signals locally on one backplane segment, globally on multiple or on all of the backplane segments. Please refer to the driver documentation for information on how to control the local buffers of the trigger bus using the software interface.

Star Trigger Bus

The first slot to the right of the system slot if reserved for the Star Trigger Controller. This slot is connected to each other slot on the backplane with an extra trigger line each. There are a total of 13 lines available. If no Star Trigger is needed within the system, the slot may be used for peripheral cards. 5 of the 13 Star Trigger lines are directly connected to the peripheral slots on the primary backplane segment. The remaining 8 Star Trigger lines are connected to the first PXI bridge. 4 lines are connected to the 7 Star Trigger lines on the secondary backplane using a multiplexer. The remaining 4 lines are routed to the second PXI bridge. The second PXI bridge connects these lines with the (up to) 7 slots of the tertiary backplane, using another multiplexer.



The structure of the multiplexers supports connections between any of the secondary or tertiary Star Trigger lines with one of the four primary Star Trigger lines. The following table gives an overview of the wiring of the Star Trigger lines.

Star Trigger line	connected with...
PXI_STAR0	PXI_STAR prim. BP, Slot 3
PXI_STAR1	PXI_STAR prim. BP, Slot 4
PXI_STAR2	PXI_STAR prim. BP, Slot 5
PXI_STAR3	PXI_STAR prim. BP, Slot 6
PXI_STAR4	PXI_STAR prim. BP, Slot 7
PXI_STAR5	PXI_STAR sec. BP Slot1 or PXI_STAR sec. BP Slot2 or PXI_STAR sec. BP Slot3 or PXI_STAR sec. BP Slot4 or PXI_STAR sec. BP Slot5 or PXI_STAR sec. BP Slot6 or PXI_STAR sec. BP Slot7
PXI_STAR6	PXI_STAR sec. BP Slot1 or PXI_STAR sec. BP Slot2 or PXI_STAR sec. BP Slot3 or PXI_STAR sec. BP Slot4 or PXI_STAR sec. BP Slot5 or PXI_STAR sec. BP Slot6 or PXI_STAR sec. BP Slot7
PXI_STAR7	PXI_STAR sec. BP Slot1 or PXI_STAR sec. BP Slot2 or PXI_STAR sec. BP Slot3 or PXI_STAR sec. BP Slot4 or PXI_STAR sec. BP Slot5 or PXI_STAR sec. BP Slot6 or PXI_STAR sec. BP Slot7
PXI_STAR8	PXI_STAR sec. BP Slot1 or PXI_STAR sec. BP Slot2 or PXI_STAR sec. BP Slot3 or PXI_STAR sec. BP Slot4 or PXI_STAR sec. BP Slot5 or PXI_STAR sec. BP Slot6 or PXI_STAR sec. BP Slot7

Star Trigger line	connected with...
PXI_STAR9	PXI_STAR tert. BP Slot1 or PXI_STAR tert. BP Slot2 or PXI_STAR tert. BP Slot3 or PXI_STAR tert. BP Slot4 or PXI_STAR tert. BP Slot5 or PXI_STAR tert. BP Slot6 or PXI_STAR tert. BP Slot7
PXI_STAR10	PXI_STAR tert. BP Slot1 or PXI_STAR tert. BP Slot2 or PXI_STAR tert. BP Slot3 or PXI_STAR tert. BP Slot4 or PXI_STAR tert. BP Slot5 or PXI_STAR tert. BP Slot6 or PXI_STAR tert. BP Slot7
PXI_STAR11	PXI_STAR tert. BP Slot1 or PXI_STAR tert. BP Slot2 or PXI_STAR tert. BP Slot3 or PXI_STAR tert. BP Slot4 or PXI_STAR tert. BP Slot5 or PXI_STAR tert. BP Slot6 or PXI_STAR tert. BP Slot7
PXI_STAR12	PXI_STAR tert. BP Slot1 or PXI_STAR tert. BP Slot2 or PXI_STAR tert. BP Slot3 or PXI_STAR tert. BP Slot4 or PXI_STAR tert. BP Slot5 or PXI_STAR tert. BP Slot6 or PXI_STAR tert. BP Slot7

The multiplexers are bi-directional. So it is possible to drive slot trigger signals from the Star Trigger controller or to trigger the star trigger controller from the peripheral cards.

Please refer to the driver documentation for information on how to control the multiplexers of the star trigger signals using the software interface.

Overview of Slot Specific Signals

The following table shows the wiring of PCI clock, IDSEL and REQ/GNT signals on the PCI Bus and the wiring of the PXI signals. Additionally it shows the geographical addresses of the slots.

	Primary Backplane								Secondary Backplane								Tertiary Backplane						
Slot No.	1	2	3	4	5	6	7		8	9	10	11	12	13	14		15	16	17	18	19	20	21
Description, Ressource	System Slot	Star Trigger Slot	Perip-heral Slot	Perip-heral Slot	Perip-heral Slot	Perip-heral Slot	Perip-heral Slot	1st PXI Bridge	Perip-heral Slot	Perip-heral Slot	Perip-heral Slot	Perip-heral Slot	Perip-heral Slot	Perip-heral Slot	Perip-heral Slot	2nd PXI Bridge	Perip-heral Slot	Perip-heral Slot	Perip-heral Slot	Perip-heral Slot	Perip-heral Slot	Perip-heral Slot	Perip-heral Slot
PCI CLK	-	P_PCI_ CLK_0	P_PCI_ CLK_1	P_PCI_ CLK_2	P_PCI_ CLK_3	P_PCI_ CLK_4	P_PCI_ CLK_5	P_PCI_ CLK_6	P_PCI_ CLK_7	P_PCI_ CLK_0	P_PCI_ CLK_1	P_PCI_ CLK_2	P_PCI_ CLK_3	P_PCI_ CLK_4	P_PCI_ CLK_5	P_PCI_ CLK_6	P_PCI_ CLK_7	P_PCI_ CLK_0	P_PCI_ CLK_1	P_PCI_ CLK_2	P_PCI_ CLK_3	P_PCI_ CLK_4	P_PCI_ CLK_5
PCI IDSEL	-	P_AD 31	P_AD 30	P_AD 29	P_AD 28	P_AD 27	P_AD 26	P_AD 25	P_AD 31	P_AD 30	P_AD 29	P_AD 28	P_AD 27	P_AD 26	P_AD 25	P_AD 24	P_AD 31	P_AD 30	P_AD 29	P_AD 28	P_AD 27	P_AD 26	P_AD 25
PCI REG / GNT	-	P_REQ/ GNT_0	P_REQ/ GNT_1	P_REQ/ GNT_2	P_REQ/ GNT_3	P_REQ/ GNT_4	P_REQ/ GNT_5	P_REQ/ GNT_6	P_REQ/ GNT_0	P_REQ/ GNT_1	P_REQ/ GNT_2	P_REQ/ GNT_3	P_REQ/ GNT_4	P_REQ/ GNT_5	P_REQ/ GNT_6	P_REQ/ GNT_7	P_REQ/ GNT_0	P_REQ/ GNT_1	P_REQ/ GNT_2	P_REQ/ GNT_3	P_REQ/ GNT_4	P_REQ/ GNT_5	P_REQ/ GNT_6
PXI_Trig	-	P_PXI_ TRIG [7:0]	P_PXI_ TRIG [7:0]	P_PXI_ TRIG [7:0]	P_PXI_ TRIG [7:0]	P_PXI_ TRIG [7:0]	P_PXI_ TRIG [7:0]	-	P_PXI_ TRIG [7:0]	P_PXI_ TRIG [7:0]	P_PXI_ TRIG [7:0]	P_PXI_ TRIG [7:0]	P_PXI_ TRIG [7:0]	P_PXI_ TRIG [7:0]	P_PXI_ TRIG [7:0]	-	P_PXI_ TRIG [7:0]	P_PXI_ TRIG [7:0]	P_PXI_ TRIG [7:0]	P_PXI_ TRIG [7:0]	P_PXI_ TRIG [7:0]	P_PXI_ TRIG [7:0]	P_PXI_ TRIG [7:0]
PXI-Star	-	-	P-PXI_STAR0	P-PXI_STAR1	P-PXI_STAR2	P-PXI_STAR3	P-PXI_STAR4	-	P-PXI_STAR0	P-PXI_STAR1	P-PXI_STAR2	P-PXI_STAR3	P-PXI_STAR4	P-PXI_STAR5	P-PXI_STAR6	-	P-PXI_STAR0	P-PXI_STAR1	P-PXI_STAR2	P-PXI_STAR3	P-PXI_STAR4	P-PXI_STAR5	P-PXI_STAR6
Geogr. Addr.	0x01	0x02	0x03	0x04	0x05	0x06	0x07		0x08	0x09	0x0a	0x0b	0x0c	0x0d	0x0e		0x0f	0x10	0x11	0x12	0x13	0x14	0x15

The indices P_..., S_...., T_.... mark the affiliation to the primary/secondary/tertiary backplane segment

The wiring of the PCI interrupt wires is compliant to the CompactPCI / PCI bridge architecture specification. See there for further details.

Connectors on Schroff PXI Backplanes

Pin Assignment PXI Connectors

Table 1: PXI System Slot 64-Bit Connector Pin Assignment

Pin	Z[14]	A	B	C	D	E	F[9]
22	GND	GA4	GA3	GA2	GA1	GA0	GND
21	GND	CLK6	GND	RSV	RSV	RSV	GND
20	GND	CLK5	GND	RSV	GND	RSV	GND
19	GND	GND	GND	SMB_SDA	SMB_SCL	SMB_ALERT#	GND
18	GND	PXI_TRIG3	PXI_TRIG4	PXI_TRIG5	GND	PXI_TRIG6	GND
17	GND	PXI_TRIG2	GND	PRST#	REQ6#	GNT6#	GND
16	GND	PXI_TRIG1	PXI_TRIG0	DEG#	GND	PXI_TRIG7	GND
15	GND	PXI_BRSVA15	GND	FAL#	REQ5#	GNT5#	GND
14	GND	AD[35]	AD[34]	AD[33]	GND	AD[32]	GND
13	GND	AD[38]	GND	V(I/O)	AD[37]	AD[36]	GND
12	GND	AD[42]	AD[41]	AD[40]	GND	AD[39]	GND
11	GND	AD[45]	GND	V(I/O)	AD[44]	AD[43]	GND
10	GND	AD[49]	AD[48]	AD[47]	GND	AD[46]	GND
9	GND	AD[52]	GND	V(I/O)	AD[51]	AD[50]	GND
8	GND	AD[56]	AD[55]	AD[54]	GND	AD[53]	GND
7	GND	AD[59]	GND	V(I/O)	AD[58]	AD[57]	GND
6	GND	AD[63]	AD[62]	AD[61]	GND	AD[60]	GND
5	GND	C/BE[5]#	GND	V(I/O)	C/BE[4]#	PAR64	GND
4	GND	V(I/O)	PXI_BRSVB4	C/BE[7]#	GND	C/BE[6]#	GND
3[3]	GND	CLK4	GND	GNT3#	REQ4#	GNT4#	GND
2[3]	GND	CLK2	CLK3	SYSEN#	GNT2#	REQ3#	GND
1[3]	GND	CLK1	GND	REQ1#	GNT1#	REQ2#	GND
25	GND	5V	REQ64#	ENUM#	3.3V	5V	GND
24	GND	AD[1]	5V	V(I/O)	AD[0]	ACK64#	GND
23	GND	3.3V	AD[4]	AD[3]	5V	AD[2]	GND
22	GND	AD[7]	GND	3.3V	AD[6]	AD[5]	GND
21	GND	3.3V	AD[9]	AD[8]	M66EN	C/BE[0]#	GND
20	GND	AD[12]	GND	V(I/O)	AD[11]	AD[10]	GND
19	GND	3.3V	AD[15]	AD[14]	GND	AD[13]	GND
18	GND	SERR#	GND	3.3V	PAR	C/BE[1]#	GND
17	GND	3.3V	IPMB_SCL	IPMB_SDA	GND	PERR#	GND
16	GND	DEVSEL#	GND	V(I/O)	STOP#	LOCK#	GND
15	GND	3.3V	FRAME#	IRDY#	GND	TRDY#	GND
14	KEY AREA						
13	KEY AREA						
12	KEY AREA						
11	GND	AD[18]	AD[17]	AD[16]	GND	C/BE[2]#	GND
10	GND	AD[21]	GND	3.3V	AD[20]	AD[19]	GND
9	GND	C/BE[3]#	GND	AD[23]	GND	AD[22]	GND
8	GND	AD[26]	GND	V(I/O)	AD[25]	AD[24]	GND
7	GND	AD[30]	AD[29]	AD[28]	GND	AD[27]	GND
6	GND	REQ#	GND	3.3V	CLK	AD[31]	GND
5	GND	BRSVP1A5	BRSVP1B5	RST#	GND	GNT#	GND
4	GND	IPMB_PWR	HEALTHY#	V(I/O)	INTP	INTS	GND
3	GND	INTA#	INTB#	INTC#	5V	INTD#	GND
2	GND	TCK	5V	TMS	TDO	TDI	GND
1	GND	5V	-12V	TRST#	+12V	5V	GND
Pin	**Z[14]**	**A**	**B**	**C**	**D**	**E**	**F[9]**

Table 2: PXI Star Trigger Slot Connector Pin Assignment

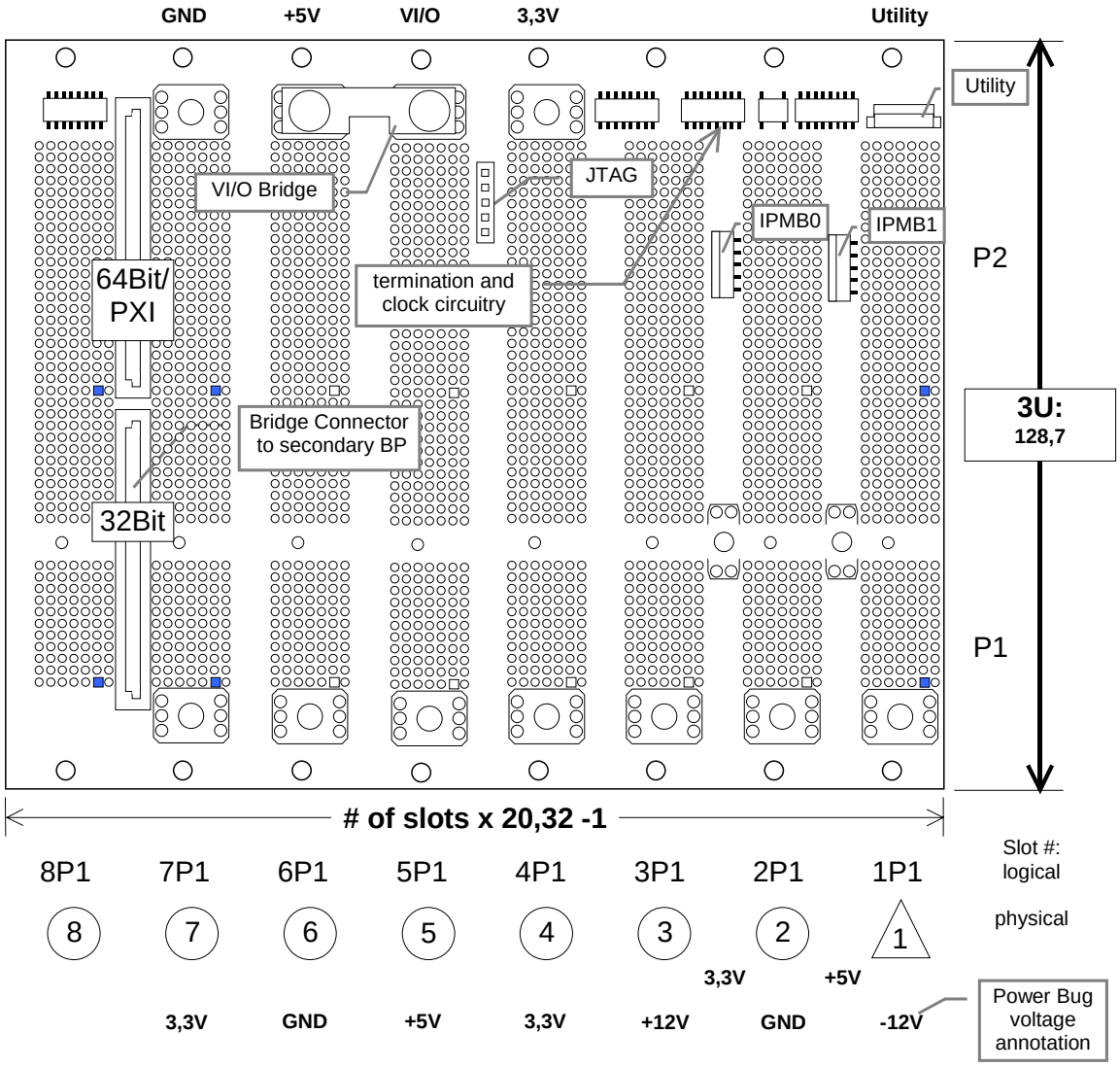
Pin	Z(14)	A	B	C	D	E	F(9)
22	GND	GA4	GA3	GA2	GA1	GA0	GND
21	GND	PXI_LBR0	GND	PXI_LBR1	PXI_LBR2	PXI_LBR3	GND
20	GND	PXI_LBR4	PXI_LBR5	PXI_STAR0	GND	PXI_STAR1	GND
19	GND	PXI_STAR2	GND	PXI_STAR3	PXI_STAR4	PXI_STAR5	GND
18	GND	PXI_TRIG3	PXI_TRIG4	PXI_TRIG5	GND	PXI_TRIG6	GND
17	GND	PXI_TRIG2	GND	RSV	PXI_CLK10_IN	PXI_CLK10	GND
16	GND	PXI_TRIG1	PXI_TRIG0	RSV	GND	PXI_TRIG7	GND
15	GND	PXI_BRSVA15	GND	RSV	PXI_STAR6	PXI_LBR6	GND
14	GND	AD[35]	AD[34]	AD[33]	GND	AD[32]	GND
13	GND	AD[38]	GND	V(I/O)	AD[37]	AD[36]	GND
12	GND	AD[42]	AD[41]	AD[40]	GND	AD[39]	GND
11	GND	AD[45]	GND	V(I/O)	AD[44]	AD[43]	GND
10	GND	AD[49]	AD[48]	AD[47]	GND	AD[46]	GND
9	GND	AD[52]	GND	V(I/O)	AD[51]	AD[50]	GND
8	GND	AD[56]	AD[55]	AD[54]	GND	AD[53]	GND
7	GND	AD[59]	GND	V(I/O)	AD[58]	AD[57]	GND
6	GND	AD[63]	AD[62]	AD[61]	GND	AD[60]	GND
5	GND	C/BE[5]#	GND	V(I/O)	C/BE[4]#	PAR64	GND
4	GND	V(I/O)	PXI_BRSVB4	C/BE[7]#	GND	C/BE[6]#	GND
3(3)	GND	PXI_LBR7	GND	PXI_LBR8	PXI_LBR9	PXI_LBR10	GND
2(3)	GND	PXI_LBR11	PXI_LBR12	UNC	PXI_STAR7	PXI_STAR8	GND
1(3)	GND	PXI_STAR9	GND	PXI_STAR10	PXI_STAR11	PXI_STAR12	GND
25	GND	5V	REQ64#	ENUM#	3.3V	5V	GND
24	GND	AD[1]	5V	V(I/O)	AD[0]	ACK64#	GND
23	GND	3.3V	AD[4]	AD[3]	5V	AD[2]	GND
22	GND	AD[7]	GND	3.3V	AD[6]	AD[5]	GND
21	GND	3.3V	AD[9]	AD[8]	M66EN	C/BE[0]#	GND
20	GND	AD[12]	GND	V(I/O)	AD[11]	AD[10]	GND
19	GND	3.3V	AD[15]	AD[14]	GND	AD[13]	GND
18	GND	SERR#	GND	3.3V	PAR	C/BE[1]#	GND
17	GND	3.3V	IPMB_SCL	IPMB_SDA	GND	PERR#	GND
16	GND	DEVSEL#	GND	V(I/O)	STOP#	LOCK#	GND
15	GND	3.3V	FRAME#	IRDY#	BD_SEL#	TRDY#	GND
14	KEY AREA						
13	KEY AREA						
12	KEY AREA						
11	GND	AD[18]	AD[17]	AD[16]	GND	C/BE[2]#	GND
10	GND	AD[21]	GND	3.3V	AD[20]	AD[19]	GND
9	GND	C/BE[3]#	IDSEL	AD[23]	GND	AD[22]	GND
8	GND	AD[26]	GND	V(I/O)	AD[25]	AD[24]	GND
7	GND	AD[30]	AD[29]	AD[28]	GND	AD[27]	GND
6	GND	REQ#	GND	3.3V	CLK	AD[31]	GND
5	GND	BRSVP1A5	BRSVP1B5	RST#	GND	GNT#	GND
4	GND	IPMB_PWR	HEALTHY#	V(I/O)	INTP	INTS	GND
3	GND	INTA#	INTB#	INTC#	5V	INTD#	GND
2	GND	TCK	5V	TMS	TDO	TDI	GND
1	GND	5V	-12V	TRST#	+12V	5V	GND
Pin	Z(14)	A	B	C	D	E	F(9)

Table 3: PXI Peripheral Slot Connector Pin Assignment

Pin	Z(14)	A	B	C	D	E	F(9)
22	GND	GA4	GA3	GA2	GA1	GA0	GND
21	GND	PXI_LBR0	GND	PXI_LBR1	PXI_LBR2	PXI_LBR3	GND
20	GND	PXI_LBR4	PXI_LBR5	PXI_LBL0	GND	PXI_LBL1	GND
19	GND	PXI_LBL2	GND	PXI_LBL3	PXI_LBL4	PXI_LBL5	GND
18	GND	PXI_TRIG3	PXI_TRIG4	PXI_TRIG5	GND	PXI_TRIG6	GND
17	GND	PXI_TRIG2	GND	RSV	PXI_STAR	PXI_CLK10	GND
16	GND	PXI_TRIG1	PXI_TRIG0	RSV	GND	PXI_TRIG7	GND
15	GND	PXI_BRSVA15	GND	RSV	PXI_LBL6	PXI_LBR6	GND
14	GND	AD[35]	AD[34]	AD[33]	GND	AD[32]	GND
13	GND	AD[38]	GND	V(I/O)	AD[37]	AD[36]	GND
12	GND	AD[42]	AD[41]	AD[40]	GND	AD[39]	GND
11	GND	AD[45]	GND	V(I/O)	AD[44]	AD[43]	GND
10	GND	AD[49]	AD[48]	AD[47]	GND	AD[46]	GND
9	GND	AD[52]	GND	V(I/O)	AD[51]	AD[50]	GND
8	GND	AD[56]	AD[55]	AD[54]	GND	AD[53]	GND
7	GND	AD[59]	GND	V(I/O)	AD[58]	AD[57]	GND
6	GND	AD[63]	AD[62]	AD[61]	GND	AD[60]	GND
5	GND	C/BE[5]#	GND	V(I/O)	C/BE[4]#	PAR64	GND
4	GND	V(I/O)	PXI_BRSVB4	C/BE[7]#	GND	C/BE[6]#	GND
3(3)	GND	PXI_LBR7	GND	PXI_LBR8	PXI_LBR9	PXI_LBR10	GND
2(3)	GND	PXI_LBR11	PXI_LBR12	UNC	PXI_LBL7	PXI_LBL8	GND
1(3)	GND	PXI_LBL9	GND	PXI_LBL10	PXI_LBL11	PXI_LBL12	GND
25	GND	5V	REQ64#	ENUM#	3.3V	5V	GND
24	GND	AD[1]	5V	V(I/O)	AD[0]	ACK64#	GND
23	GND	3.3V	AD[4]	AD[3]	5V	AD[2]	GND
22	GND	AD[7]	GND	3.3V	AD[6]	AD[5]	GND
21	GND	3.3V	AD[9]	AD[8]	M66EN	C/BE[0]#	GND
20	GND	AD[12]	GND	V(I/O)	AD[11]	AD[10]	GND
19	GND	3.3V	AD[15]	AD[14]	GND	AD[13]	GND
18	GND	SERR#	GND	3.3V	PAR	C/BE[1]#	GND
17	GND	3.3V	IPMB_SCL	IPMB_SDA	GND	PERR#	GND
16	GND	DEVSEL#	GND	V(I/O)	STOP#	LOCK#	GND
15	GND	3.3V	FRAME#	IRDY#	BD_SEL#	TRDY#	GND
14	KEY AREA						
13	KEY AREA						
12	KEY AREA						
11	GND	AD[18]	AD[17]	AD[16]	GND	C/BE[2]#	GND
10	GND	AD[21]		3.3V	AD[20]	AD[19]	GND
9	GND	C/BE[3]#	IDSEL	AD[23]	GND	AD[22]	GND
8	GND	AD[26]	GND	V(I/O)	AD[25]	AD[24]	GND
7	GND	AD[30]	AD[29]	AD[28]	GND	AD[27]	GND
6	GND	REQ#	GND	3.3V	CLK	AD[31]	GND
5	GND	BRSVP1A5	BRSVP1B5	RST#	GND	GNT#	GND
4	GND	IPMB_PWR	HEALTHY#	V(I/O)	INTP	INTS	GND
3	GND	INTA#	INTB#	INTC#	5V	INTD#	GND
2	GND	TCK	5V	TMS	TDO	TDI	GND
1	GND	5V	-12V	TRST#	+12V	5V	GND
Pin	Z(14)	A	B	C	D	E	F(9)

Schroff PXI Backplanes

8 Slot, rear view, System Slot left



Avaiable Backplanes and Accessories

PXI Backplanes 3U, 64-bit, System Slot left

Article Number	Slot Count	Description	V I/O	Bus frequency
23006-575	5	primary	+5V	33 / 66 MHz
23006-577	7	primary	+5V	33 MHz
23006-578	8	primary	+5V	33 MHz
23006-587	7	secondary	+5V	33 MHz
23006-594	4	tertiary	+5V	33 MHz
23006-597	7	tertiary	+5V	33 MHz

PXI Bridge

Article Number	Bus width / Orientation	Bus frequency	Description
23006-924	32-bit / left to right	33 / 66 MHz	Low profile

Mechanical and Climatic Parameters

	Backplanes	Bridges
Operating Temperature	0°C - +70°C	
Storage Temperature	-55°C - +105°C	- 45°C - +70°C
Humidity	max 95%, not condensing Conformal Coating (on request)	
Flammability PCB, Connectors Ceramic caps	 UL 94 V-0 fire-proof	
Connectors Performance level per IEC 61076-4-101 Mechanical Durability (Mating Cycles) Total Insertion and Extraction Force (mating)	IEC 61076-4-101 (HardMetric 2mm Grid) level 2 (level 1 on request) >250 cycles (> 500 cycles on request) < 0,75 N / Pin	
Vibration acc. DIN 41640 Part 15	10Hz – 500Hz (5Hz – 2000Hz on request) 5g rms (20g rms on request)	
Shock (10 pulses each direction x,y,z)	10g, 6ms	
Low Pressure / Altitude (max Board voltage per single isolation gap doesn't exceed 12V)	no restrictions	
Construction	12 - Layer Stripline	
Dimensions (mm) Width (pl. see Dwg.) Height 3U / 6U Thickness	 20,32mm x #Slots-1mm 128,7mm / 262,05mm 3,9mm +/- 0,2mm	 50mm 100mm (64-bit), 60mm (32-bit) 10mm

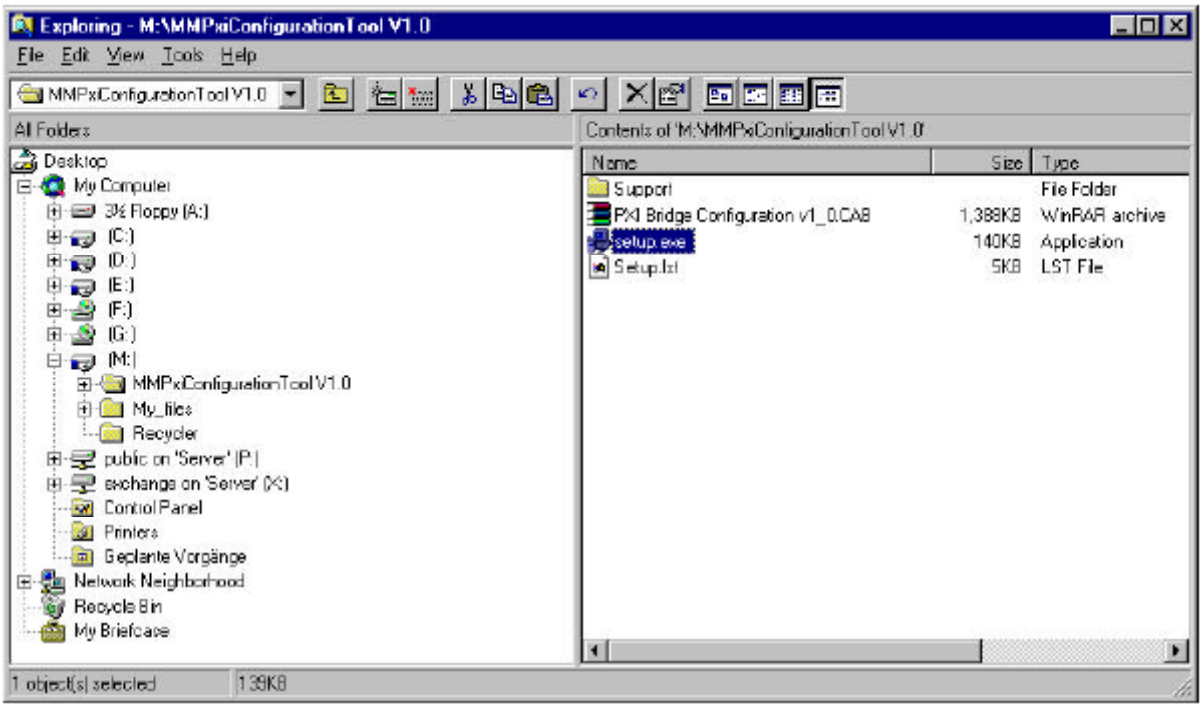
Electrical Parameters

Specifications	PICMG 2.0 R3.0 CPCI Core Specification PICMG 2.1 CPCI Hot Swap Specification PICMG 2.6 Bridging Specification PICMG 2.8 Pin Registration for PXI PICMG 2.9 System Management Bus Spec. PICMG 2.10 Keying Specification PXI Spec PXI Specification Rev. 2.0
Service Life: MTBF acc. to MIL HDBK 217F, cond.: 25°C, ground, benign 3U 8-Slot	more than 600.000h
Characteristic Impedance PCI traces PXI Local Bus traces	65 Ω ± 10 % 75 Ω ± 10 %
Ohmic Resistance of Signal Traces PCI traces PXI Local Bus traces	< 80mΩ/Slot < 90mΩ/Slot
Hot Swap	supported
Termination (only 8 Slot Backplanes)	Schottky diodes (on request), plugable termination board
Power Input	Power bugs for wiring or special Adapter Board to use an ATX cable; this board can act as a power distribution star point within the Systems
max. Current carrying Capacity 5V/GND 3,3V/GND	8 A per Slot 10 A per Slot
max. Voltage Drop between any two points on the backplane on +5V or +3,3V	< 40mV
VI/O bridging (default)	+5V (default), blue key; 3,3V optional (yellow key) field changeable, using M4 screws and a bus bar (fixed during bp assy by using a Power Bug cable using Faston crimp contacts on request)
PCI Clock Frequency	33 MHz, 66 MHz up to 5 Slots; on higher Slot number M66EN can by enabled for test purposes (cut a copper link on rear)
PCI Bus Width	64bit
Data Transfer Rate (peak) 33 MHz 66 MHz	132 Mbyte/s (32 bit) / 264 Mbyte/s (64 bit) 264 Mbyte/s (32 bit) / 528 Mbyte/s (64 bit)
Bridging of Backplanes	backplane of slot numbers equal or higher than 4 up to 7 Slots can be bridged. Special secondary and tertiary backplanes available.
PXI Clock Accuracy Switching between external and internal sources Min Pulse width Min. time between successive edges of the same polarity:	10 MHz Skew: < 0,5 ns; < 1,0 ns for bridged BP´s, Jitter: < 0,2 ns > 30 ns > 80 ns

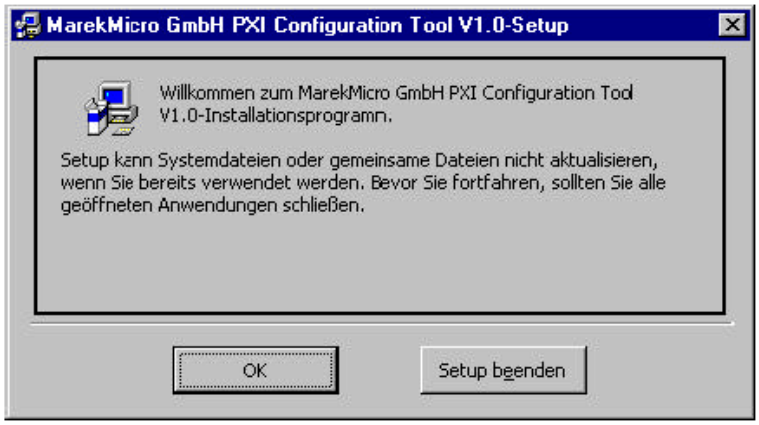
Software Installation, PXI Bridge (for Windows 2000)

Installation of the Software Interface

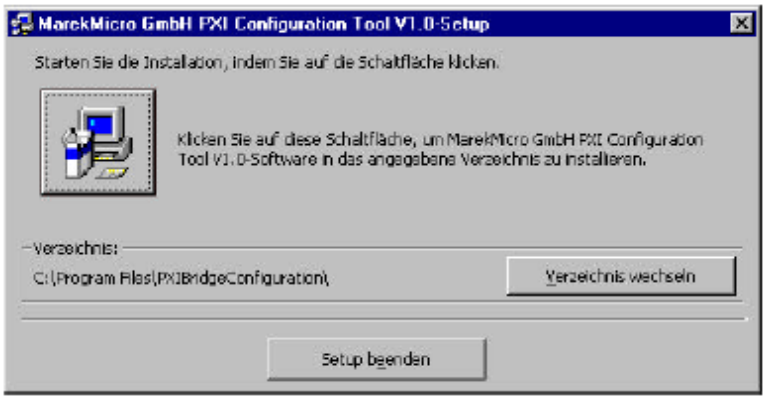
To install the software on your system, double click setup.exe



In the next window you find general information. Confirm by clicking “OK”.



The following window looks like this:



Click the large button (the computer symbol) to continue installation. By clicking “Verzeichnis wechseln” you can change the installation folder.

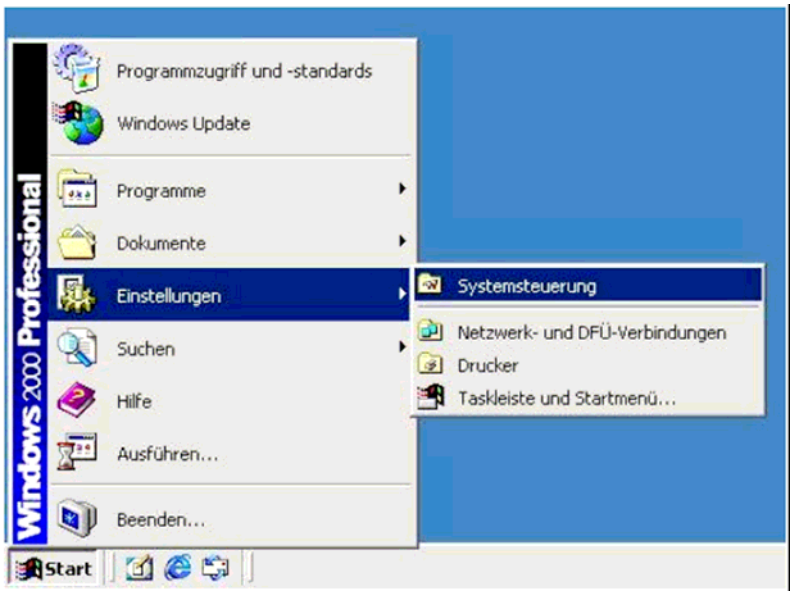
Setup creates a new program group called “MarekMicro GmbH” and inserts the element “PXI Bridge Configuration”. So the PXI Bridge Configuration Software can later be accessed through the Windows Start Menu. You can also choose a different group name or select a name from the list “Vorhandene Gruppen”. Continue by clicking “Weiter”.

Setup now copies the required files to the target folder. The process finishes with this message:

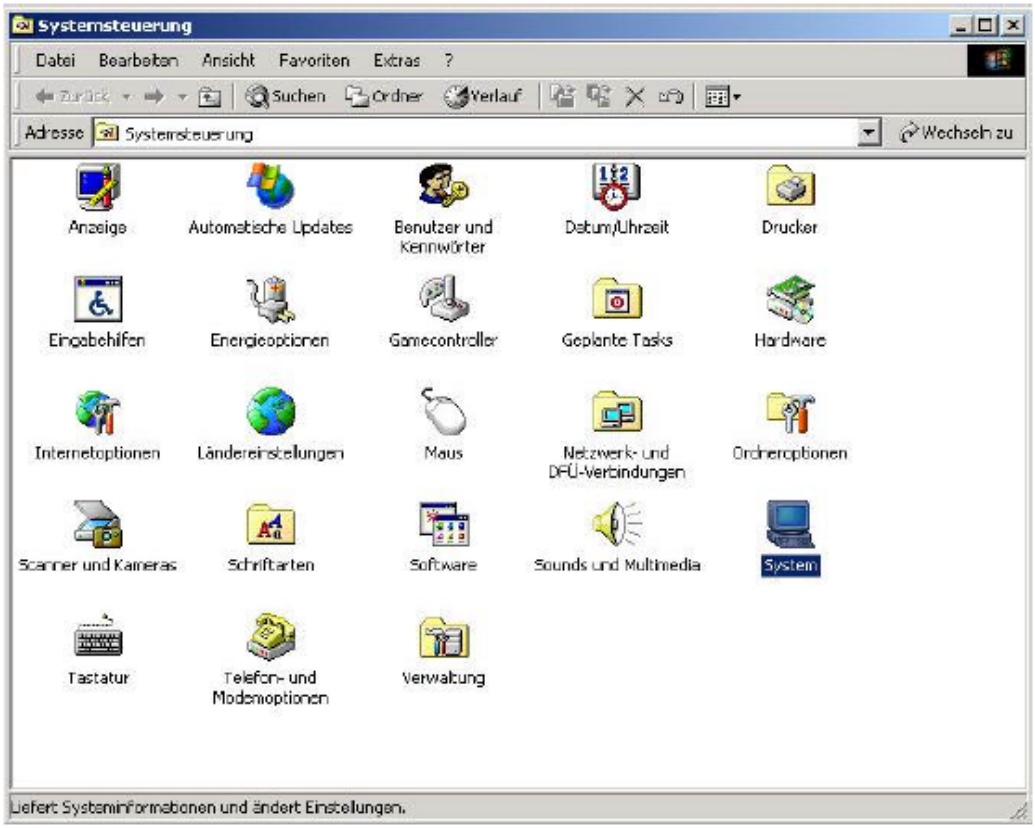
Click “OK” to finish the installation of the graphical interface.

Driver Installation

The standard driver installed by Windows 2000 does not cover the necessary functions. As Windows 2000 only installs a new driver when adding new hardware, the driver for the PXI bridge needs to be installed manually. Open the Control Panel (Start -> Settings -> Control Panel)



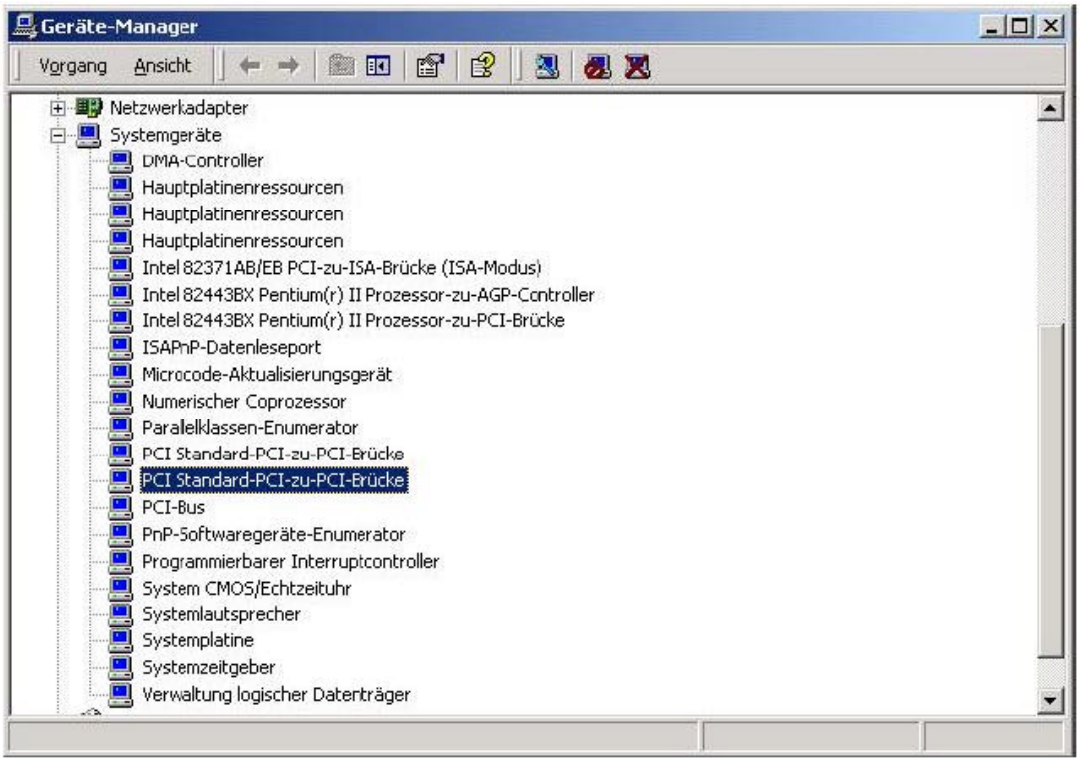
Double click „System“.



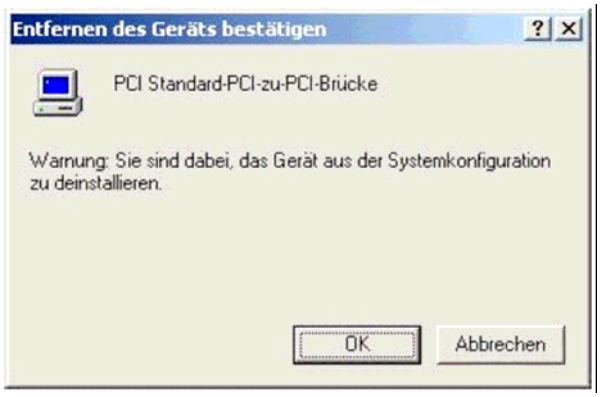
Choose the register “Hardware” and click “Device Manager”.

Now check the tree named “System Devices” for entries “PCI Standard-PCI-to-PCI-Bridge” and remove these entries by hitting the delete key on your keyboard.

Usually it is OK to only remove the uppermost entry as the following entries will then also be removed.



In the next window confirm by clicking the “OK” button and restart your system when asked, so Windows can load the driver for the PXI bridge.



After rebooting the plug and play manager finds the “new” hardware.

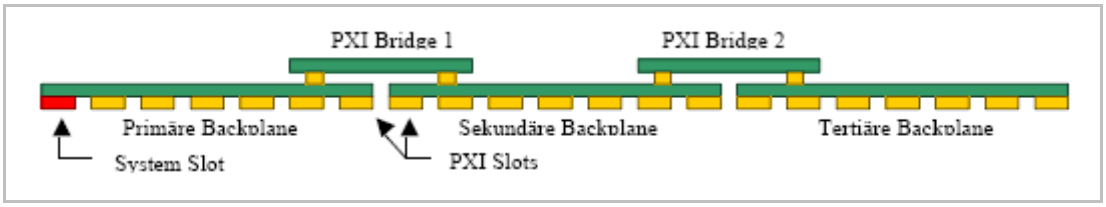
Now the correct driver will be installed.

After this the installation of the user interface and the driver is complete.
By default you can find the configuration software in the Windows Start Menu at:
“Start -> Programs -> MarekMicro GmbH -> PXI Configuarion Tool”.

PXI Bus Configuration Tool

Introduction

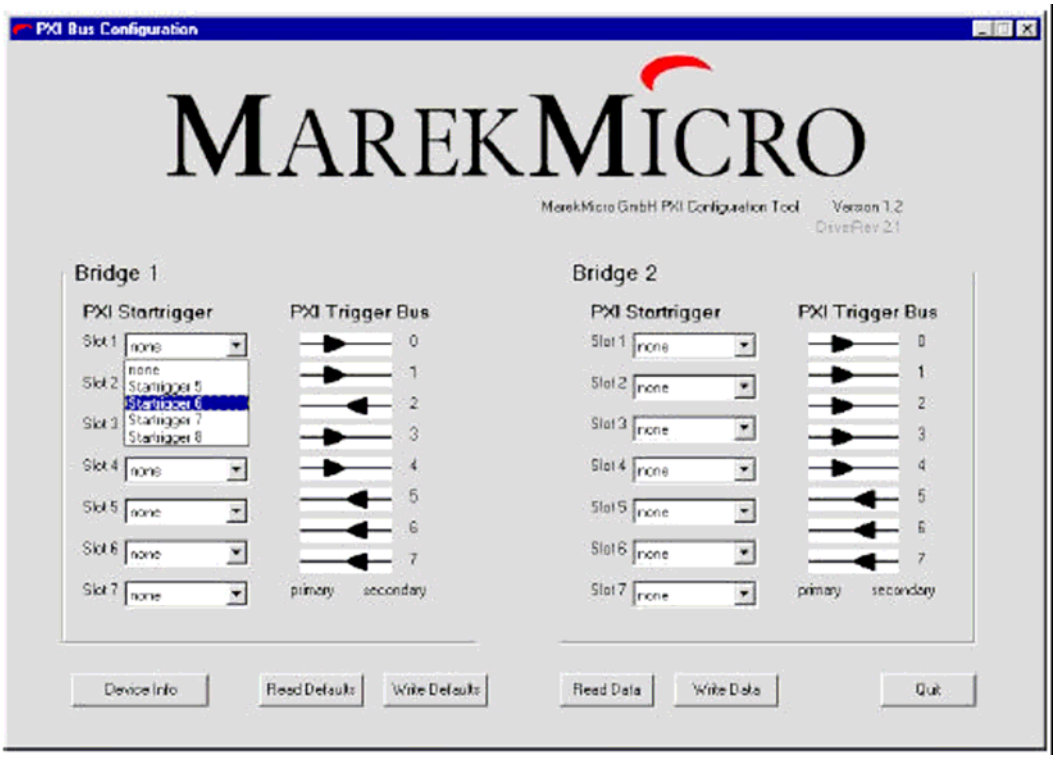
Schroff PXI Bridges allow cascading of several backplane segments. The first backplane segment, the primary backplane, contains a system slot for the CPU and six peripheral slots. All other backplane segments only contain up to seven peripheral slots for use with PXI modules. The maximal usage with 21 slots looks like this:



Use the PXI Bus Configuration Tool to configure the PXI bridges. It supports systems with one or two PXI bridges.

Description of the User Interface

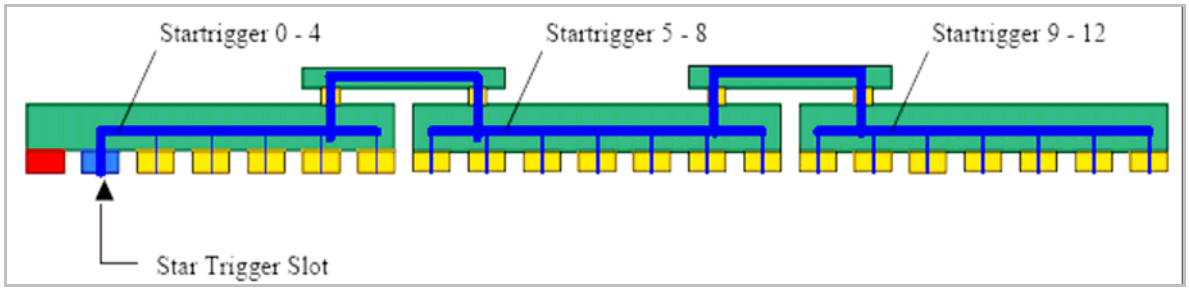
In this illustration you see the PXI Bus Configuration Tool with up to two bridges and their configuration state.



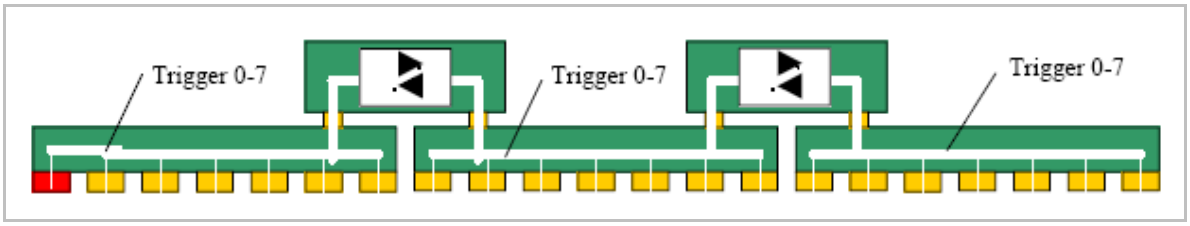
The possible setting for both PXI bridges are identical, so in this description we only consider one bridge.

The left column of a bridge shows the PXI star trigger signals, which can be assigned to any slot.

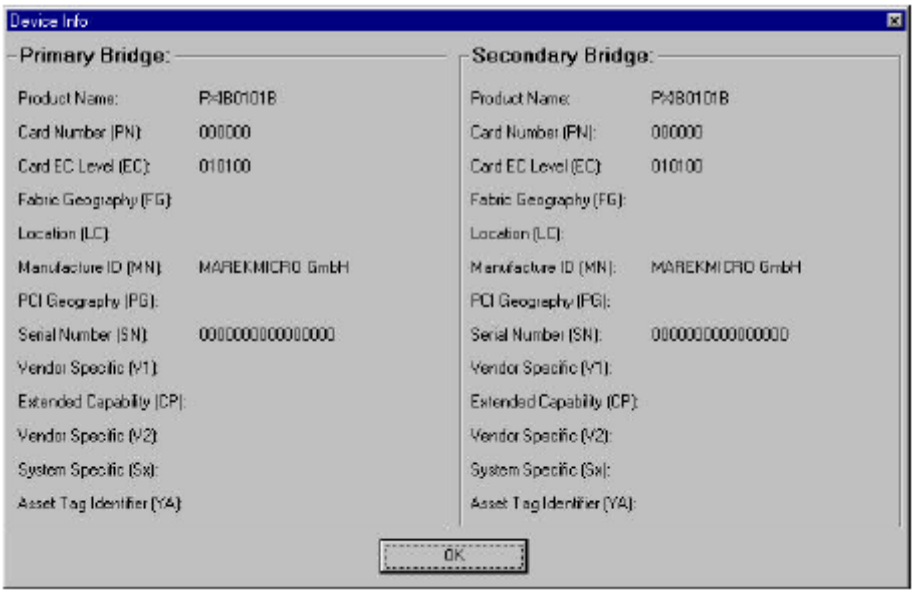
Of all the 13 star trigger signals, 5 lead to the slots to the right of the star trigger slot, which is slot number 2 on the primary backplane. 4 signals are connected to the secondary backplane (star triggers 5-8) and further 4 signals are connected to the tertiary backplane (star triggers 9-12). You can connect any of the seven slots of the secondary or tertiary backplane with one of those 4 available star trigger signals.



In the right column there are the bi-directional and buffered PXI trigger bus signals. Connections in both directions can be used. By clicking any of the signals, an arrow pointing to the secondary side occurs. If a trigger signal to the opposite direction is needed, the signal has to be clicked another time. It is not possible to buffer a trigger signal in both directions.



The modified setting have to be confirmed by “Write Config”. By “Read Config” the current settings of the bridge(s) can be read. By “Write Defaults” the current settings are written to the system registry. At every system restart those settings are sent to the bridge. By “Read Defaults” you can load the saved settings to the configuration interface. “Device Info” shows VPD (Vital Product Data) of the bridge:



To exit the PXI Bridge Configuration Tool, click “Quit”.

Explanations to the entries in “Device Info”:

- “Product Name” shows the exact designation of the product.
- “Card EC Level (EC)” shows the current revision
- “Manufacturer ID (MN)” shows the vendor name
- The serial number of the bridge can be found at “Serial Number (SN)”
- The remaining entries are currently not used by Schroff but can be read

If there is just one bridge, only the left half of the menu is shown while the right half is invisible.

Description of the Driver Interface

General Information about the Driver

This section describes the features of the device driver PXI_BRIDGE for the bridge module, which works as connection unit of a bridged PXI backplane with up to 21 slots. The supported operating system is Windows 2000 SP3 or newer.

Knowledge of the basic features and structure of this operating system are required as well as basic knowledge of Win32 API and the programming language “C”. A description of the PXI_BRIDGE hardware can be found in this manual, previous pages.

Basic Information about the Kernel Mode Driver

The basic purpose of a kernel mode driver is to encapsulate details of the supported hardware and provide the application (user mode application) with an abstracted calling interface. With Windows 2000, the link between the user mode program and the kernel mode driver is created by the Win32 API, especially the IO manager. For communication between driver and application the functions *CreateFile(), OpenFile(), ReadFile(), WriteFile(), DeviceIOControl()* and *CloseFile()* are used. By calling *CreateFile()* the application gains access to the selected hardware. *CreateFile()* needs to be called for each installed module in order to gain access. At each access only one module can be addressed. If several modules are to be served sequentially, a respective amount of driver calls is necessary.

Registry Entries

Following keys must be entered in the target system’s registry. If the Schroff PXI Bridge Configuration Software and driver are installed as described in this manual, these entries are already existing.

```
[HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\pxiBridge]
"ErrorControl" = dword:00000001
"Type" = dword:00000001
"Start" = dword:00000002
"Group" = "PCI Configuration"
"DisplayName" = "pxiBridge"

[HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\pxiBridge\parameters]
"Bridge1Config"=dword:0
"Bridge2Config"=dword:0
```

The upper block defines settings for the driver, the lower block defines two parameters for the registry. These are two 32 Bit values containing the configuration of the PXI Bridges. Bridge1Config contains the configuration of the first bridge. This is the bridge sitting closer to the system slot. Bridge2Config contains the configuration data of the second bridge - if applicable. At driver startup (automatically at system start) those two configuration values are written to the respective PXI bridge. They can be changed by the PXI Configuration Tool. A Bit description of both values can be found at „PXIBRIDGE_IOCTL_WRITE_GPIO“.

Driver Interface

CreateFile()

This function is used for opening the driver. The function call supplies a handle to the device with the Win32 name \\.\PXI_BRIDGE. This handle is to be used by all succeeding driver calls.

Example:

```
HANDLE DriverHandle;
DriverHandle = CreateFile("\\\\.\\PXI_BRIDGE", GENERIC_READ | GENERIC_WRITE,
0, NULL, OPEN_EXISTING, FILE_ATTRIBUTE_NORMAL, NULL);
if(DriverHandle==INVALID_HANDLE_VALUE) {
MessageBox("Cannot Open Driver ", "Error", MB_ICONINFORMATION | MB_OK);}
```

OpenFile()

This function is not supported by the PXI bridge driver.

ReadFile()

This function is not supported by the PXI bridge driver.

WriteFile()

This function is not supported by the PXI bridge driver.

CloseFile()

This function is used by the application to close the driver. After calling this function no more accesses by the application are possible.

Example:

```
HANDLE DriverHandle;
DriverHandle = CreateFile(....);
...
...
CloseFile(DriverHandle);
```

DeviceIOControl()

This function is used by the application for all data transfers. DeviceIOControler supports handing over subroutines as well as data buffers for read and write data.

Example:

```
#include "pxiBridge.h”
...
BOOL IoctlReturn; // Return code;
UINT IBuf, OBuf; // Input and Output Buffer
ULONG ReturnSize; // Number of bytes placed into the output buffer by the driver.
IoctlReturn = DeviceIoControl(
DriverHandle, // Handle to driver
PXIBRIDGE_IOCTL_GET_DRIVER_REV, // Function number
&IBuf, // Address of Input-Buffer
sizeof(UINT), // Length of Input-Buffer
&OBuf, // Address of Output-Buffer
sizeof(UINT), // Length of Output-Buffer
&ReturnSize, // Nmb of Bytes written to Output-Buffer
0); // overlapped structure = 0
```

After calling the driver, *IoctlReturn* contains information about the operation being successful or not. A value of zero means error, all values different from zero mean the execution was successful. At each call of *DeviceIOControl* the handle returned by *CreateFile()* is to be supplied. The functions supported by *DeviceIOControl* are defined in the file “pxiBridge.h”. For input and output buffer both start address and length of the buffer are given. If one of the buffers is not used, the start address is to be set to the value NULL and the length to the value 0. *ReturnSize* contains the number of bytes to be written to the output buffer by the driver.

PXIBRIDGE_IOCTL_GET_DRIVER_REV

This function returns the driver version. Bit[31:16] Major Version, Bit[15:0] Minor Version

Example:

```
#include " pxiBridge.h”
...
BOOL IoctlReturn; // Return code;
ULONG ReturnSize; // Size of returned bytes.
ULONG dat; // Dummy variable
IoctlReturn = DeviceIoControl(
DriverHandle, // Handle to driver
PXIBRIDGE_IOCTL_GET_DRIVER_REV, // Function number
NULL, // Address of Input-Buffer
0, // Length of Input-Buffer
& dat, // Address of Output-Buffer
```

```
sizeof(ULONG), // Length of Output-Buffer
&ReturnSize, // No. of Bytes written to Output-Buffer
0); // overlapped structure = 0
....
printf ("PXIBRIDGE Driver Revision; %d\n", dat);
...
```

PXIBRIDGE_IOCTL_GET_NMB_OF_BRIDGES

This function returns the number of PXI bridges installed in the system.

Example:

```
#include " pxiBridge.h”
...
BOOL IoctlReturn; // Return code;
ULONG ReturnSize; // Size of returned bytes.
ULONG dat; // Dummy variable
IoctlReturn = DeviceIoControl(
DriverHandle, // Handle to driver
PXIBRIDGE_IOCTL_GET_NMB_OF_BRIDGES, // Function number
NULL, // Address of Input-Buffer
0, // Length of Input-Buffer
& dat, // Address of Output-Buffer
sizeof(ULONG), // Length of Output-Buffer
&ReturnSize, // No. of Bytes written to
// Output-Buffer
0); // overlapped structure = 0
....
printf ("Number of PXI bridges installed; %d\n", dat);
```

PXIBRIDGE_IOCTL_WRITE_GPIO

This function writes the PXI_BRIDGE configuration to the hardware. A struct is given as parameter in the input buffer, which contains a 32 Bit value with all settings for each bridge. If there is only one bridge in the system, the driver only uses the configuration value for bridge one

Structure of the struct:

```
typedef struct _BR_CONFIG
{
ULONG ConfigBridge1 ; // 32 Bit Config Value for Bridge 1
ULONG ConfigBridge2; // 32 Bit Config Value for Bridge 2
} BR_CONFIG, *P_ BR_CONFIG;
```

Bitfields of the values in BR_CONFIG

ConfigBridge1	
Byte 0 / 1	Activation and selection of star triggers, 4 Bits for each trigger (5, 6, 7, 8) Bit 0 Enable Bit 1..3 Slot number 1..7 = (0..6)
Byte 2	Trigger Bus „to secondary“ 0..7 enable
Byte 3	Trigger Bus „to primary“ 0..7 enable

ConfigBridge2	
Byte 6 / 7	Activation and selection of star triggers, 4 Bits for each trigger (9, 10, 11, 12) Bit 0 Enable Bit 1..3 3 Slot number 1..7 = (0..6)
Byte 2	Trigger Bus „to secondary“ 0..7 enable
Byte 3	Trigger Bus „to primary“ 0..7 enable

Example:

```
#include " pxiBridge.h”
...
BOOL brCfgReturn; // Return code;
BR_CONFIG brCfg; // Structure
ULONG returnSize; // Size of returned bytes.
ULONG dat; // Dummy variable
brCfg.ConfigBridge1= xxx; // Config Value for Bridge 1
brCfg.ConfigBridge2= xxx; // Config Value for Bridge 2
IoctlReturn = DeviceIoControl( DriverHandle, // Handle to driver
PXIBRIDGE_IOCTL_WRITE_GPIO, // Function number
&brCfg, // Address of Input-Buffer
sizeof(BR_CONFIG), // Length of Input-Buffer
NULL, // Address of Output-Buffer
0, // Length of Output-Buffer
&ReturnSize, // Bytes returned
0); // overlapped structure = 0
....
```

PXIBRIDGE_IOCTL_READ_GPIO

This function reads the bridge configuration data from the hardware and supplies a 32 Bit value for the configuration software. A struct is passed as parameter to the output buffer, containing a 32 Bit value with all settings for each bridge. If there is only one bridge in the system, the driver only writes the configuration value for bridge one

```
typedef struct _BR_CONFIG
{
ULONG ConfigBridge1; // 32 Bit Config Value for Bridge 1
LONG ConfigBridge2; // 32 Bit Config Value for Bridge 2
} BR_CONFIG, *P_ BR_CONFIG;
```

Example:

```
#include " pxiBridge.h”
...
BOOL brCfgReturn; // Return code;
BR_CONFIG brCfg; // Structure
ULONG returnSize; // Size of returned bytes.
ULONG dat; // Dummy variable
IoctlReturn = DeviceIoControl( DriverHandle, // Handle to driver
PXIBRIDGE_IOCTL_READ_GPIO, // Function number
NULL, // Address of Input-Buffer
0, // Length of Input-Buffer
&brCfg, // Address of Output-Buffer
sizeof(BR_CONFIG), // Length of Output-Buffer
&ReturnSize, // Bytes returned
0); // overlapped structure = 0
....
// Access the data read
dat = brCfg.ConfigBridge1;
```

PXIBRIDGE_IOCTL_SAVE_DEFAULT

This function saves the configuration data of both PXI bridges to the registry.
A struct is passed as a parameter in the input buffer, containing a 32 Bit value with all settings for each bridge. If there is only one bridge in the system, the driver only uses the configuration value for bridge one

```
typedef struct _BR_CONFIG
{ ULONG ConfigBridge1; // 32 Bit Config Value for Bridge 1
ULONG ConfigBridge2; // 32 Bit Config Value for Bridge 2
} BR_CONFIG, *P_ BR_CONFIG;
```

Example:

```
#include " pxiBridge.h”
...
BOOL brCfgReturn; // Return code;
BR_CONFIG brCfg; // Structure
ULONG returnSize; // Size of returned bytes.
USHORT dat; // Dummy variable
brCfg.ConfigBridge1= xxx; // Config Value for Bridge 1
brCfg.ConfigBridge2= xxx; // Config Value for Bridge 2
IoctlReturn = DeviceIoControl( DriverHandle, // Handle to driver
PXIBRIDGE_IOCTL_SAVE_DEFAULT, // Function number
&brCfg, // Address of Input-Buffer
sizeof(BR_CONFIG), // Length of Input-Buffer
NULL, // Address of Output-Buffer
0, // Length of Output-Buffer
&ReturnSize, // Bytes returned
0); // overlapped structure = 0
....
```

PXIBRIDGE_IOCTL_GET_DEFAULT

This function loads the configuration data of both PXI bridges from the registry. A struct is passed as parameter in the output buffer, containing a 32 Bit value with all settings for each bridge. If there is only one bridge in the system, the driver only uses the configuration value for bridge one

```
typedef struct _BR_CONFIG
{ ULONG ConfigBridge1; // 32 Bit Config Value for Bridge 1
ULONG ConfigBridge2; // 32 Bit Config Value for Bridge 2
} BR_CONFIG, *P_ BR_CONFIG;
```

Example:

```
#include " pxiBridge.h”
...
BOOL brCfgReturn; // Return code;
BR_CONFIG brCfg; // Structure
ULONG returnSize; // Size of returned bytes.
ULONG dat; // Dummy variable

IoctlReturn = DeviceIoControl( DriverHandle, // Handle to driver
PXIBRIDGE_IOCTL_GET_DEFAULT, // Function number
NULL, // Address of Input-Buffer
0, // Length of Input-Buffer
&brCfg, // Address of Output-Buffer
sizeof(BR_CONFIG), // Length of Output-Buffer
&ReturnSize, // Bytes returned
0); // overlapped structure = 0
....
// Access the data read
dat = brCfg.ConfigBridge1;
```

PXIBRIDGE_IOCTL_READ_VPD

This function reads VPD information from both bridges. The driver supplies a 472 (2*236) bytes large array containing VPD information of both bridges. The values for bridge one start at offset 0, the values for bridge two start at 236. If there is only one bridge in the system, the driver only supplies VPD information for bridge one, the area for bridge 2 is empty (0x00)

Example:

```
#include " pxiBridge.h”
...
BOOL IoctlReturn; // Return code;
UCHAR vpd[2*VPD_SIZE]; // Array of bytes
ULONG ReturnSize; // Size of returned bytes.
IoctlReturn = DeviceIoControl( DriverHandle, // Handle to driver
PXIBRIDGE_IOCTL_READ_VPD, // Function number
NULL, // Address of Input-Buffer
0, // Length of Input-Buffer
&vpd, // Address of Output-Buffer
2*VPD_SIZE, // Length of Output-Buffer
&ReturnSize, // Bytes returned
0); // overlapped structure = 0
if (IoctlReturn == 0)
printf("PXIBRIDGE_IOCTL_READ_VPD: failed\n");
else
{
....
Access vpd information
```

Driver Revision History

Revision 0.01

First revision for debug purposes.

Revision 0.02

General improvement of first revision.

Revision 0.03

Additions.

Revision 0.04

Added VPD support.