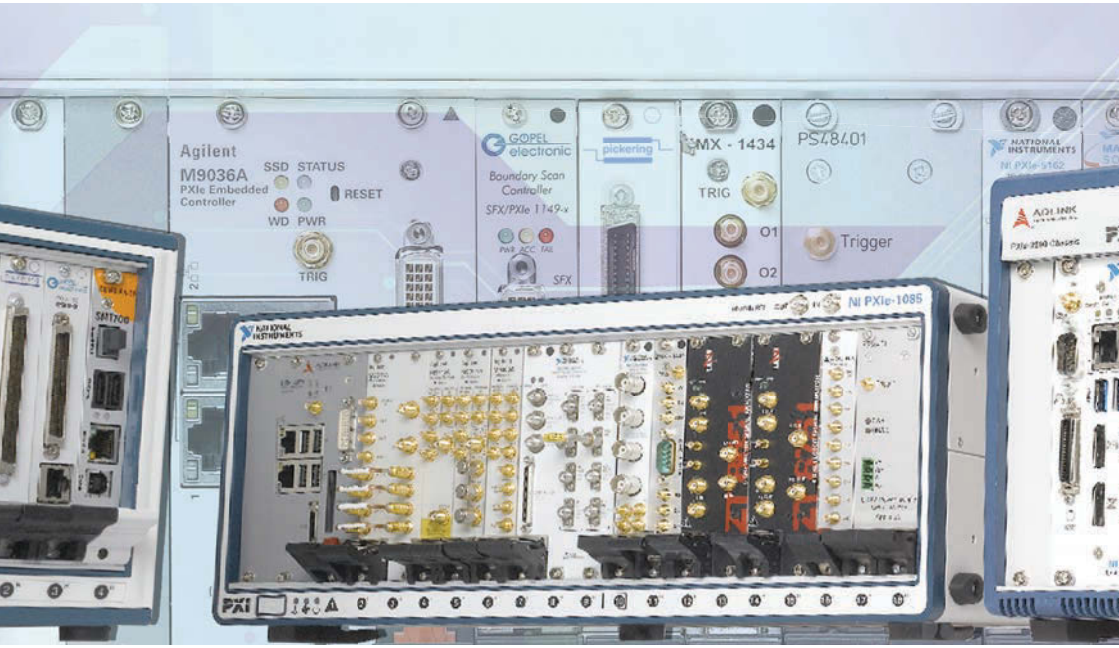
PCI eXtensions for Instrumentation

PXImate

A practical guide to using PXI



5th Edition

pickering

PXImate

This book provides an overview of the PXI standard and its derivative versions.

It provides an overview for those new to PXI systems and a useful source of reference material for the more experienced.

This is a living document that Pickering Interfaces will continue to develop in support of the PXI standard and its future evolution. We welcome any feedback from users on subjects they would like to be included in future issues.

Please note that, while this document adheres technically to the PCI and PXI Specifications, statements on the status of various segments of the PXI market represent the opinion of Pickering Interfaces and not that of the PXISA.

PXI Dates:

August 1997	Revision 1.0 of the PXI Standard published
July 2000	Revision 2.0 of the PXI Standard published
February 2003	Specification separated and published as Software and Hardware Revision 2.1
September 2003	VISA for PXI specification published
August 2005	PXI Express (PXIe) Hardware and Software specification published
September 2009	PXI Multicomputing (PXImc) specification published, followed by installer in September 2010
October 2012	PXI and PXIe Trigger Management specification published.

CONTENTS

SECTION 1

INTRODUCTION TO PXI BASICS

BACKGROUND AND HISTORY

PXI is a modular instrument system designed to take advantage of the fast data interfaces based on PCI and PCIe bus systems. The standard is an open standard that any vendor can use in creating a product and the standard is designed to ensure that modules from different vendors will operate in any vendor's chassis.

The PXI standard defines the mechanical, electrical and software interfaces provided by PXI compliant products, ensuring that integration and software costs are minimized and allowing trouble-free multi-vendor solutions to be implemented.

In use a PXI system appears as an extension to the PCI or PCIe slots in the user's controller regardless of whether the controller is embedded in the PXI Chassis or is a separate computer.

In 2005 the standard expanded to cover two physical implementations of the PCI bus, namely PCI (later often referenced as Classic PCI) and PCIe. These two versions of the bus are largely software compatible but are not mechanically or electrically compatible. The two versions are referenced as PXI and PXIe where PXI uses the multi-drop parallel bus structure of PCI and PXIe uses the point to point serial interface of PCIe. Chassis can be designed that support both control methods in the same physical slot to provide support for either style of module. A commonly used abbreviation PXI(e) is often used to indicate that a statement applies to both PXI and PXIe.

The physical form factor is based on the cPCI standard but with the addition of connections used to support triggering functions and on PXI, a local bus.

The PXI modules providing the instrument functions are plugged into a chassis which may include its own controller or a PCI(e) to PXI(e) bridge that provides a high speed link from a PC.

Most PXI instrument modules are simple register based products that use software drivers to configure them as useful instruments, taking advantage of the increasing power of computers to improve hardware access and simplify embedded software in the modules. The control model uses a central controller to provide "intelligence" through the software drivers which must be compatible with the operating system of the system controller.

CompactPCI and PXI modules are interchangeable - they can be used in either CompactPCI or PXI Chassis - but installation of PXI modules into a cPCI chassis removes any ability that a PXI module has to support the dedicated hardware triggers and local bus of PXI. However, in practice many PXI modules do not support the hardware triggers or use the Local Bus.

Since the introduction of the PXI Standard and the PXIe version two additional options have been added - PXImc (PXI MultiComputing) and a Trigger Bus Management system for the PXI(e) Chassis. Neither of these has seen much adoption so far but the trigger bus management is very recent and has strong interest from several vendors.

PXI CHASSIS BASICS

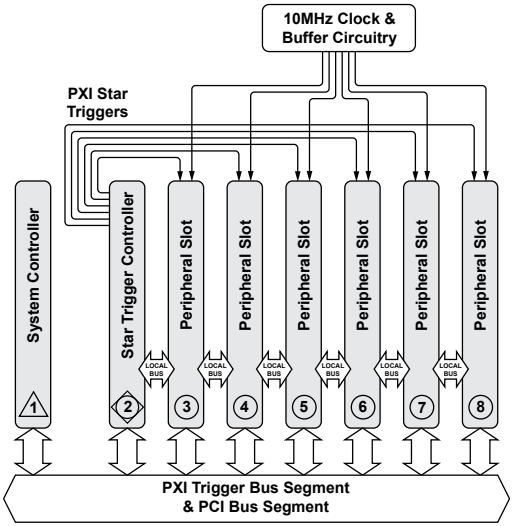


Fig. 1.1 - 8 Slot PCI Bus configuration

The PXI Chassis provides the mechanical means of mounting the PXI modules and providing the forced air cooling to the modules. It also provides DC power, the PCI bus and the PXI specific functions. The chassis are typically designed to house either 3U or 6U PXI modules, the 6U modules being twice the height of the 3U modules. In general 3U modules can be fitted into 6U Chassis using an adaptor. The PXI standard supports the design of chassis that allow both 3U and 6U modules to be used. The 3U size though is by far the dominant module size.

The chassis allows CompactPCI modules to be added, but CompactPCI specific features are not supported.

The PXI specification supports the use of 32-bit and 64-bit PCI backplane connections at 33MHz and 66MHz bus speeds, ensuring theoretical bus speeds of 132 Mbytes/sec to 528 Mbytes/sec respectively, a speed far in excess of that available over GPIB or typical VXI interfaces.

The specification allows up to 8 slots on each 33MHz PCI segment, or 5 slots per 66MHz PCI segment. This does not limit the number of slots available in the chassis, since larger chassis include PCI to PCI Bridges to interconnect segments. Each PCI Bridge occupies one electrical slot on each of the segments it connects to.

Many modules only support 33MHz operation and the same is true of PXI Chassis - the increased complexity of a 66MHz backplane (requiring more bridges so as to not exceed the bus segment limit) means the speed increase comes at a cost and

conveys no great advantage to the user since speed of the backplane is rarely a limiting factor in test systems. Virtually all PXI modules only support 32-bit connection and not 64-bit connections, and the standard even permits the chassis to only support 32-bit connections.

For 33MHz systems one PCI Bridge conveniently supports a total of 14 slots in a chassis (a Slot 1 and 13 Peripheral Slots) - two PCI ports are occupied by the PCI Bridge (one on each bus segment).

Chassis based on 66MHz are much less common since the numbers of available slots per PCI segment are lower - a 14 slot chassis would need three PCI Bridges instead of one which increases cost. In addition the presence of a single module not supporting 66MHz operation will automatically limit the backplane to 33MHz. There are very few PXI modules designed for 66MHz (or 64-bit) operation.

Most chassis include a 64-bit bus but the standard does permit a 32-bit only implementation that has to be stated in the data sheet. The motive for this is usually in portable applications where power consumption can be reduced.

Most implementations of PXI systems are therefore 32-bit at 33MHz with both modules and the chassis limiting the BW to reduce cost and consequently do not reach the headline speeds often quoted for PXI.

The presence of PCI Bridges should generally be transparent to the user of a PXI Chassis. However, if two modules require the exchange of trigger signals over the Trigger Bus additional complications will arise because the Trigger Bus does not directly cross the PCI Bridge. The Star Trigger is wired to cross the first PCI Bridge but has a limited connection count.

The PCI Bridges introduce a one clock delay in transferring information from one segment to another.

If an instrument requires the use of two separate modules connected by the trigger bus the instrument's operation can be complicated or disrupted if its modules are inserted either side of a PCI Bridge. It is therefore best to avoid dividing these modules with a bridge. There is a new document that standardizes the software that controls any trigger bus links which may be present, but the trigger bus links are not mandated and neither is their functionality.

The location of a PCI Bridge is marked on the chassis and backplane by a vertical line shown between the slot numbers.

Fig. 1.2 - Chassis PCI Bridge glyphs

(image shown - Pickering's 40-923A-001)

The PXI specification does not set out a rigorous standard for what can be included in a PXI Chassis, though all must comply with the mandatory parts of the specification. For that reason PXI Chassis vary in their capability and the user needs to choose the chassis that is right for the application. Things to consider are:

- Number of modules required. Too much capacity makes the chassis larger and more expensive, too little forces the use of more than one chassis.
- The module sizes (3U and 6U) required in the system. If 6U modules are required as well as 3U a mixed size chassis may be needed, one that supports both module heights. It should be noted that a 3U PXI Module will plug into a 6U slot and work perfectly but may require a mechanical adaptor. Some chassis may permit dual stacking of 3U modules - 2 off 3U modules in a single 6U slot.
- Diagnostics support to check that power supplies and fans continue to operate correctly.
- Power supply capacity. Too little power could prevent the inclusion of high power consumption modules. Some test systems may require more power on specific voltage supplies, for example some analogue or RF functions may require higher currents on the ±12 V supplies than systems that only test logic circuits. Chassis designed in accordance with Version 2.1 of the specification may provide more power than chassis designed against earlier versions of the standard. There is also a low power version of the specification that can be used where the chassis power is reduced compared to the full power specification.

- Fan air capacity affects cooling rate within PXI modules, and influences the maximum power dissipation in each module. A fan speed controller can reduce acoustic noise at normal temperatures and help reduce temperature changes in the test system, but in practice the improvements are not major. If a chassis is required to work in an office environment and the modules do not generate a large thermal load a chassis with lower cooling capacity and lower acoustic noise may have some advantages. Fan air capacity indicates the capacity of the fans and not necessarily how much air is moved through the chassis – something that can be dependent on the modules installed as different modules present different air flow resistance.
- A built-in display can help monitor test progress, but takes up space in the chassis. A built-in monitor may help in the design and development phase, but be less relevant once the test system is deployed in an automatic test environment. Only a small number of applications tend to include displays in the chassis.
- Inclusion of additional drives, such as CDs or DVDs, for directly loading programs or storing data. Again this is not common in PXI(e) systems.
- Support of trigger bus connections across the bus segments if hardware triggering is essential.

Having provided an outline description of the PXI Chassis the following sections provide a little more detail.

PXI Slot Numbers

Each PXI slot has an associated slot number which is marked (in most cases) below the PXI slot. They typically number left to right.

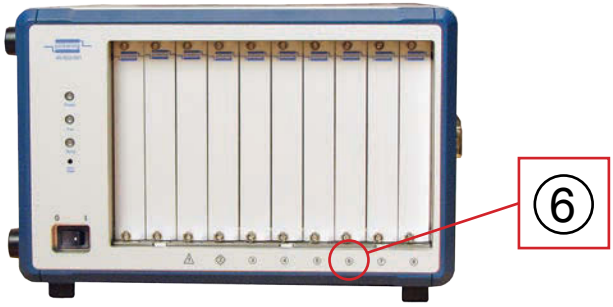


Fig. 1.3 - A small PXI Chassis with slot numbers below each slot

(image shown - Pickering's 40-922-001)

6U Chassis and 3U Module Stacking

Some 6U Chassis permit 3U modules to be “stacked”, two 3U modules per 6U slot. The stacking arrangement typically is confined to a few slots as shown below.

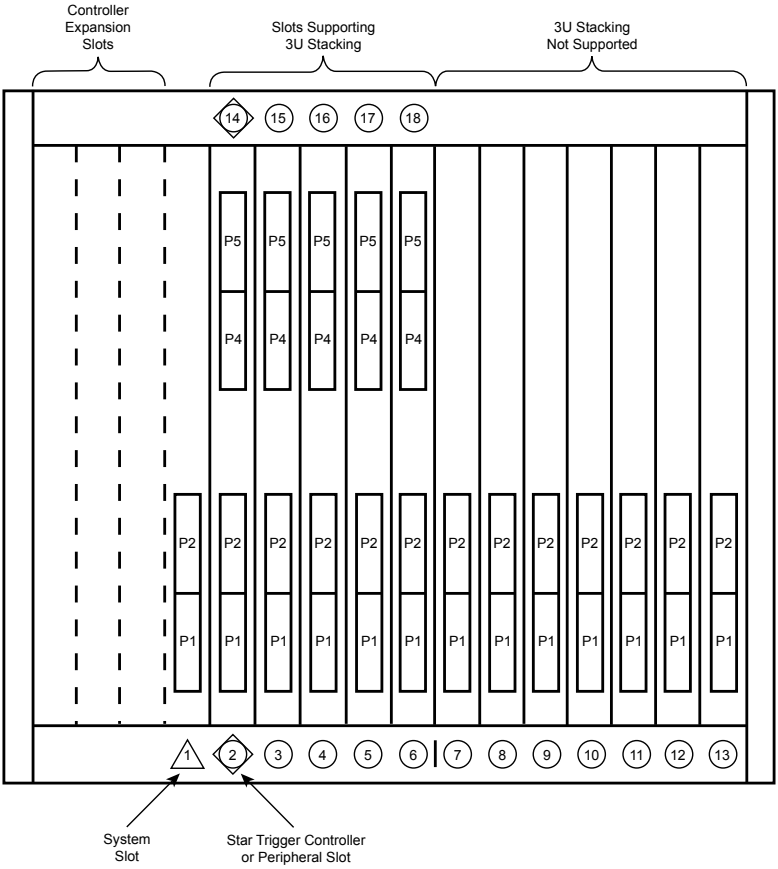


Fig. 1.4 - Slot numbering for a 6U Chassis supporting 3U stacking

A 6U Chassis supporting stacking will have the P1 and P2 modules that support 3U PXI modules in the bottom position and then the dual stack connectors P4 and P5 will support a second 3U PXI module in the top of the slot. Slot numbers for the top PXI modules appear above the top 3U module, those without stacking capability will have no slot number above them.

PXI Backplane

The PXI backplane is typically a single PCB constructed from multiple layers (it is often a very thick PCB because of the high layer count and the use of power planes for power distribution) into which the PXI modules are plugged. It provides all the control and power signals to the modules.

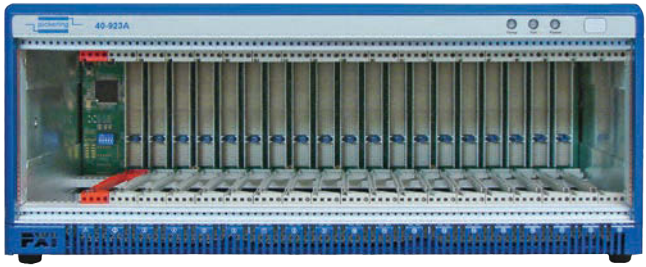


Fig. 1.5 - 19 Slot Chassis showing backplane
(image shown - Pickering's 40-923A-001)

PXI Bus and Enumeration

Most PXI systems will have a 33MHz 32-bit wide bus connecting the modules together in a series of bus segments separated by PCI Bridges. The PCI Bridges are "transparent" to the user software.

The PCI bus uses a process referred to as Bus Enumeration to allocate a physical PXI slot with a programmatically defined identity. For PCI there are two identifications, the PCI Bus Segment and the Device Number within that segment. As PCI uses a multi-drop system each bus segment (the bus between PCI Bridges) has a Bus Number and within that Bus Number up to 16 devices can be supported, referenced as the Device Number. The physical location of the device is "hard wired" into the chassis. So a Peripheral Module installed in the chassis in a particular slot has a Bus Number and a Device Number associated with it. The slight complication is that the Bus Number is determined by a Bus Enumeration algorithm and two controllers may not allocate a particular bus segment with the same Bus Number, for example if one controller has a PCI graphics card and the other does not bus numbering may be done differently, and if a controller has a new PCI part added that will change the numbering. PCI buses internal to the controller generally get counted first and then the enumeration works out from the controller. This complication is handled by VISA and other software tools and is largely invisible to the user.

PCI has an absolute number of buses it can support – 256 – and this applies to PXI as well. This is not a significant limitation as each bus will typically have up to 7 devices attached to it when in a PXI system.

The PXI backplane provides power to the PXI modules and provides the PCI interface. The backplane has to include any PCI Bridges that are required to maintain the integrity of the PCI interface. In addition the backplane delivers other PXI features, such as triggering, Local Bus and Star Trigger, described elsewhere in this book.

Chassis Power

The PXI standard specifies the minimum power that can be delivered to each module. The following minimum specification applies for chassis conforming to Version 2.1 of the PXI specification.

	System Slot	Peripheral slot	System Slot	Peripheral slot	All	All
Power Supply Voltage	+5 V	+5 V	+3.3 V	+3.3 V	+12 V	-12 V
Current requirement	6 A	2 A	6 A	2 A	0.5 A	0.25 A

Fig. 1.6 - Chassis Power Supply minimum requirements

The chassis power supply must be capable of supplying the stated current on each supply rail with all the available slots populated and fully loaded. The table sets the minimum average current the chassis should provide, it does not limit the current a single module can draw - there are additional requirements covering this aspect.

The chassis power supply and backplane must be able to deliver 1 Amp to any individual Peripheral Slot on the +12 V and -12 V supplies and any Peripheral Slot must be allowed to draw 6 Amps on the +5 V supply. It should be noted that chassis built to versions of the standard earlier than 2.1 do not necessarily conform to the above requirements.

If a chassis is fitted with many modules that have a high current consumption it is possible to exceed the power supply rating, but this is rarely the case in in most test systems since many modules have current consumption well below the average ratings.

Each pin on each backplane connector passes a maximum of 1 Amp. Each connector uses multiple pins to pass the required current. In accordance with the above table, the +5V and +3.3V supplies each require 6 pins to be connected. Many chassis are capable of providing more current at each voltage. This information should be provided in the chassis manual.

System Reference Clock

The PXI backplane always provides a system reference clock operating at 10 MHz and having an accuracy of 100 ppm (parts per million) or better. The clock is specified to have a 50% ±5% duty cycle and each module is independently driven to avoid module interaction. The reference is distributed such that each slot receives the signal at the same time to within 1 ns.

For some applications the system reference clock is not accurate or stable enough, particularly for RF applications where it may define the frequency accuracy of an RF carrier. The Star Trigger slot has a specific pin assigned to allow it to provide the

alternative Reference Clock. The PXI specification recommends that the backplane provides a facility to switch the clock to the alternative frequency source provided by the Star Trigger slot (slot 2).

Local Bus

The Local Bus is a daisy chain of 13 lines that interconnects adjacent PXI slots. Each Local Bus line on the right of the slot is connected to the neighboring Local Bus slot on its left. The bus is used to allow adjacent modules to exchange analogue signals (up to ±42 V) or digital signals directly. Software has to check the compatibility of adjacent slots before the modules can make use of the facility. If modules that make use of the Local Bus are not placed in the appropriate positions then the Local Bus functionality may be lost, since there is no requirement for modules to provide a bridge, and different modules will use the Local Bus in different ways (or more typically not at all).

The slot located next to the system controller (Slot 2 Star Trigger location) is a special case. The Local Bus lines on the left side (facing the controller) are dedicated to the Star Trigger function and do not connect to the System Controller.

The bandwidth and other characteristics of the local bus are not fixed in the standard, limiting how designers can use these connections.

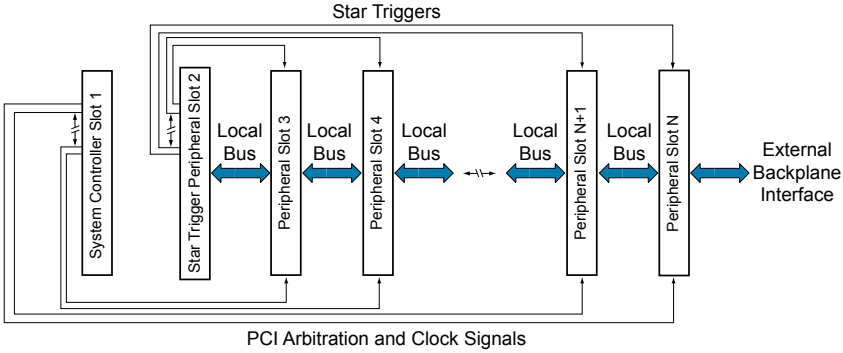


Fig. 1.7 - PXI Local Bus routing

Trigger Bus

The 8 PXI triggers provide a method of synchronizing the operation of PXI modules. The bus is configurable so that any module can send triggers to other modules in the system and respond to events in other modules. The standard does not mandate the buffers or the support tools required.

The Trigger Bus is not permitted to cross a PCI Bridge without the addition of a buffering system in chassis to electrically isolate the segments. The Trigger Bus length is also limited to a distance of 252 mm. As the standard does not mandate the buffers and the support tools needed these may not be included in the chassis and some may simply

use manually configured jumpers. For this reason it is recommended that modules using the Trigger Bus to trigger events are placed on the same Trigger Bus segment. The PCI Segment Divider Glyph on the front of the chassis clearly identifies the location of any PCI Bridge.

The 8 PXI triggers can provide a low latency method of triggering events - it is possible for the triggers to be routed directly to the hardware. However in most PXI implementations trigger operations do involve the driver (and in the case if IVI all involve the driver) so actual speed of operation is defined by software latency.

Star Trigger

The Star Trigger is a high performance trigger designed to provide a high speed trigger line between the first Peripheral Slot (Slot 2) and the other Peripheral Slots. The modules in these Peripheral Slots must support the Star Trigger lines in order for the feature to work as defined here. The Star Trigger makes use of the lines on the left hand side which would normally be part of the Local Bus. The Star Trigger slot is not an essential part of a chassis, but in practice most chassis provide it. A Star Trigger slot can be used as an ordinary Peripheral Slot if a Star Trigger is not required.

The position of the Star Trigger is marked on the backplane and the module by the Star Trigger Slot Glyph shown.

Fig. 1.8 - Star Trigger Slot glyph

When a Star Controller is installed in the chassis it can ensure that events in other peripherals are simultaneously triggered with very low levels of differential time delay between modules, through the use of the Star Trigger. The trigger system is bidirectional so the Star Trigger module can allow an event in one Peripheral Module to trigger events in other modules.

The Star Controller provides thirteen output lines, each going to a different designated Peripheral Module. In a 14 slot chassis there are 12 Peripheral Slots (14 slots less Slot 1 and Slot 2) and typically one PCI Bridge (for a 33 MHz PCI bus). The thirteen Star Triggers are connected to the 12 Peripheral Slots. For a larger chassis the higher slot numbers (beyond physical slot 15) do not support the Star Trigger, they should be occupied by modules not requiring this facility.

For systems requiring more than one chassis, fitting a Star Trigger Module into each chassis and connecting them together, either through cables, or synchronizing them through the use of GPS timing, ensures that PXI instrumentation systems can perform more complex measurements in distributed systems.

As with the 8 PXI triggers Star Trigger operation may be executed through the driver, in which case synchronization may be dependent on software latency.

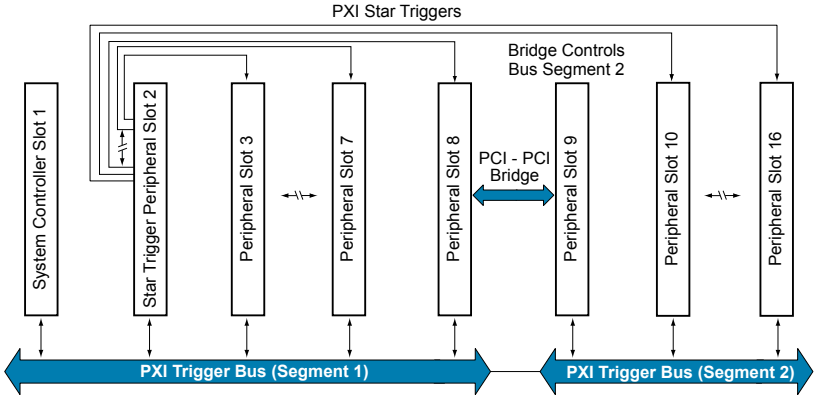


Fig. 1.9 - PXI architecture for two Bus Segments

PXI MODULES

The PXI standard defines the mechanical outline and connectors for both 3U and 6U modules.

The 3U modules have one ejector handle fitted at the bottom of the module. A screw fixing is provided at the top and the bottom, the bottom fixing being partly hidden by the ejector handle. Modules occupying more than one slot can have more than two screw fixings. Two connectors can be fitted (J1, J2), but the module may have no functions requiring the use of the J2 connector (64-bit PCI and PXI features) and for that reason it may be omitted from the module to save cost.

The 6U modules have two ejector handles fitted and two connectors (J1, J2). Any other connectors fitted (defined as J3, J4, J5) are outside the scope of the standard and may create mechanical and electrical compatibility issues. Fixing screws are used at the top and bottom of the module, partly hidden by the ejector handles. Modules occupying more than one slot can have more than two fixing screws.

It is good practice that modules are operated with all fixings tightly secured, this is particularly important on modules that require a good earth connection. The front panel ground must be isolated from the PXI power supply ground to avoid ground loop currents. The module performance should be specified with the screws tightened.

In recent years as PXI has entered new application areas for which the restricted front panel space is an issue, there has been a tendency for some modules to drop the ejector handle.

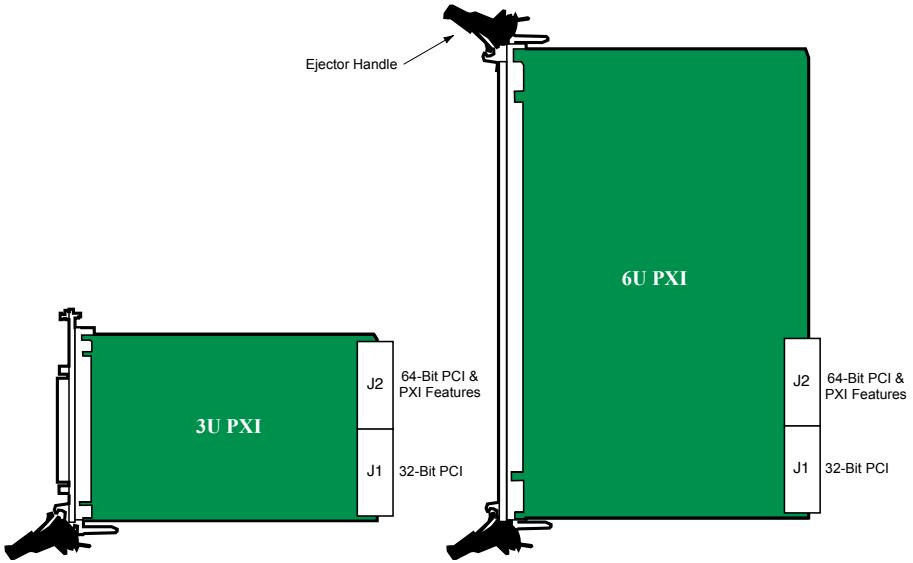


Fig. 1.10 - PXI Peripheral Module form factors and connectors

In addition to offering PXI modules with specific instrument functionality, manufacturers also offer prototyping PXI modules which provide a PXI interface, some control lines and other useful features. These modules allow users to create their own special purpose modules without the need to invest in creating the PXI framework needed to make it operate in the PXI Chassis. Once the hardware is created drivers can be written that turn this module into their own PXI building block.

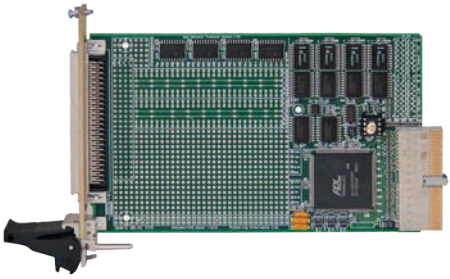


Fig. 1.11 - Typical Prototyping Module

(image shown - Pickering's Breadboard Module 40-220)

PXI Slot 1, System Slot

The left hand slot of the chassis is reserved for the system controller. The slot has one set of connectors to the backplane and occupies one notional slot. However, in reality a controller may require more than one slot and for this reason the standard allows the chassis to expand the Slot 1 position to the left hand side (away from the rest of the Peripheral Modules). A typical chassis may have three Controller Expansion Slots, allowing up to the equivalent of four module widths to be occupied by the controller, but using just one set of backplane connectors. The Controller Expansion Slots do not have backplane connectors.

The PXI Chassis can use either an embedded controller in the Slot 1 position, or an interface module allowing connection to an external controller (such as a PC). The interface module is typically based on a proprietary serial interface sent over a copper or optical cable connected between the controller and the module in the PXI Chassis. The interface behaves as a PCI to PCI Bridge and is therefore transparent to the system software. A PCI interface card is required in the controller to provide the serial output and a PXI interface module in the chassis decodes the data and transfers it to the PCI bus. For example, National Instruments' MXI-3 uses a proprietary 1.5 Gb/s serial link as the interconnection system, ADLINK use four open standard Star Fabric serial interfaces that provide an aggregate 2.5 Gb/s serial link to perform the same function. It is also possible to use External PCIe systems.

The same interface modules can be used to expand the number of chassis in the system. An additional PXI interface module is installed in any of the available slots of the first chassis and connected via the serial interface to an identical PXI interface module in the Slot 1 position of the second chassis.

SECTION 2

PXI Express

PXI EXPRESS

As the demands on PC speed grew the speed of the PCI bus increasingly became an issue, the concept of a multi-drop system based on a parallel bus structure becoming ever harder to scale to escalating PC performance requirements.

A breakthrough was made with the introduction of a fast serial interface, PCIe, which carries data on wire pairs usually referred to as PCIe lanes. A single lane is no faster than a PCI 64-bit 33MHz interface, however lanes could be aggregated into higher data rate connections, four lanes being a popular early configuration. The serial connection is also point to point, so a particular connection only carries traffic destined for the node connected at the end of it (and in any connections downstream of that device) and has no un-terminated transmission line stubs to distort the high speed waveforms. That allowed the speed of each lane to be increased as technology improved. This serial interface system is inherently more scalable than a parallel bus system.

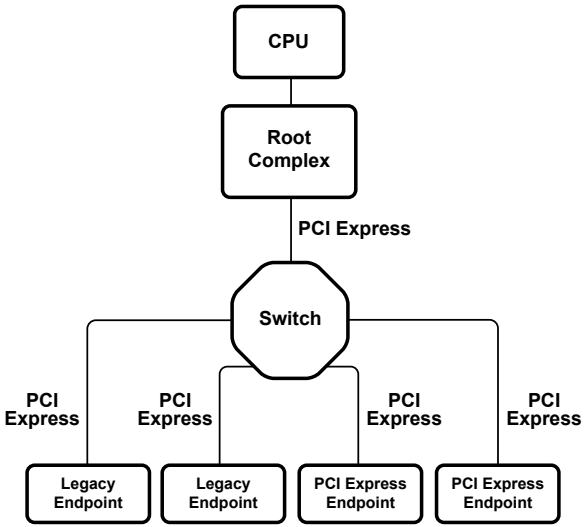


Fig. 2.1 - PCI Express

PCIe was introduced at GEN1 with a raw bit rate of 2.5Gb/s (2Gb/s after decoding), subsequent generations have increased that data rate and implementations have increased the numbers of lanes to provide greater data bandwidths. Mechanisms are provided that transparently cause the data rate to drop if a lower rate (by GEN number or lane count) device is connected downstream of a higher speed connection.

The data connection rate is chassis, chassis slot and module dependent and as always the greater the data rate the more the cost of implementation. Other than this, users are virtually unaware of the data management processes going on at the PCIe interface.

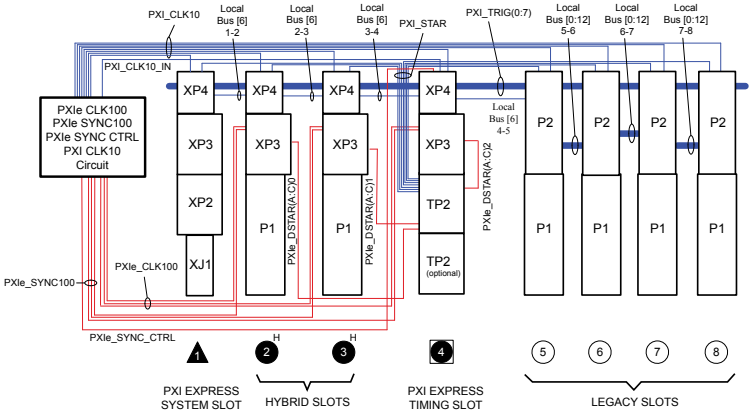
The structure is a tree style, a single PCIe connection expands into multiple connections below it which in turn can branch into further connections. For connections at the trunk of the tree (Root Complex) there is a need for high BW to maximize the data capacity since it has to support all the end points on the downstream side.

As with PCI all traffic flows to and from the Root Complex, actual speed is dependent both on the PCIe interface and the ability of the controller to handle all the data and drivers. PCIe was added to the PXI standards as PXIe. As with PXI and PCI there were extensions required to add test measurement features in creating a PXIe standard.

PXIe Bus Enumeration

The bus enumeration on a PXIe Chassis is a little different to PXI Chassis. In PXI the location is defined by a Bus Segment and a Bus Device because each Bus Segment can support multiple Peripheral Modules. In PXIe, the End Points are simply one device, so essentially there is a Bus Segment for every device connected and additional Bus Segments for connecting busses. The consequence of this is that PXIe systems inherently have a much higher number of Bus Segments than PXI systems, in some cases this can lead to problems because controllers designed for fast boot time may not fully enumerate deep PCIe bus systems. For this reason PXI vendors often recommend a limited range of controllers for use on PXIe, the range being model specific not just PC vendor specific. A PC vendor may have different models with differing enumeration capabilities.

PXIe Chassis



P1	Connector carrying 32 Bit PCI Bus (PXI)
P2	Connector carrying upper 32 Bit PCI Bus (PXI), Trigger, Local Bus
XP2, XP3	Connector carrying PXI Express Interface
XP4	Connector equivalent to partially populated P2
XJ1	PXI Express System Slot connector
TP2	PXI Express Timing Slot connector (TP1 is optional)

Fig. 2.2 - PXIe Backplane

A PXIe Chassis uses a similar mechanical principle to a PXI Chassis, but the backplane and the connection to the PXIe modules is different both electrically and mechanically. A chassis may include support for both PXIe and PXI modules as discussed in a later section of this publication. For simplicity, the following describes a pure PXIe Chassis and concentrates on the 3U format rather than the 6U format.

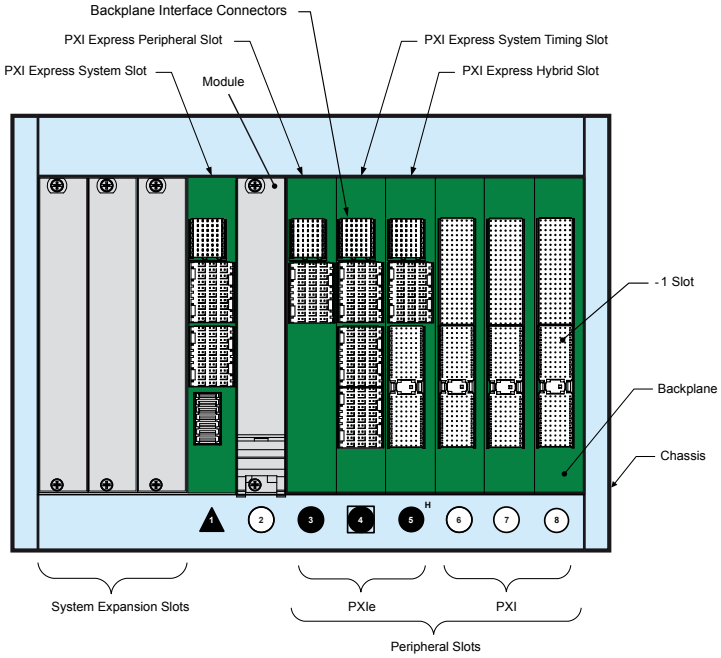


Fig. 2.3 - A PXI Express Chassis showing slot types and a module

The glyphs underneath each slot are visually different to PXI so that there is a clear indication this is a PXIe slot, PXIe uses a dark background with white slot numbering whereas PXI uses a light background with dark slot numbering.

Backplane

A PXIe backplane uses PCIe connections rather than PCI connections to provide a control interface. The PCIe connection can essentially be of any PCIe generation (GEN1, GEN2, GEN3) and can have a differing number of lanes connected to each slot. This is again an important difference from PXI, not all slots are the same. A user who has a module that demands high data connection speeds needs to choose the slot it is placed in to maximize the data bandwidth. If a “low” data rate module is placed in a high data rate slot the PCIe will adjust the data bandwidth to suit the module, if a “high” data rate module is placed in a “low” data rate slot then the module will be serviced at the slot data rate.

Some chassis include a means of configuring the backplane lane connections to increase their flexibility. In particular they allow lanes to be concentrated into a few interfaces so that data hungry modules can be serviced more rapidly and given greater bandwidth than other slots.

There are many other differences in the detailed implementation of the backplane on PXIe, for example the use of 100MHz clock in addition to 10MHz clock. The trigger system is based on point to point differential signalling rather than multi-drop single ended signalling. Only one Local Bus is supported to connect adjacent modules together, so vendors have removed their dependency on the local bus.

Chassis Power

The backplane carries the chassis power to the modules. The PXIe Chassis supports two power rails rather than the four in PXI, +3.3V and +12V, for the Peripheral and Timing Slot modules and is required to support +5V for the System Controller Slot. The table below shows the minimum power that the chassis must deliver to the slots.

Table 2.1 - PXIe Chassis power

Slot Type	3.3V		12V		5V	Maximum Slot Power
	Minimum backplane	Minimum PSU	Minimum backplane	Minimum PSU		
System Controller	9A	9A	11A	11A	9A	
PXIe only slot or Timing Slot	6A	3A	4A	2A	0	30W

Notes:

- System Controller currents are for 3 or more slots occupied.
- The chassis power supply must supply the minimum current on each supply to each of the slot types at the same time – so the chassis power supply must be capable of supplying at least a system controller slot plus an amount equal to the number of PXIe/timing slots multiplied by the minimum PSU current for each supply.
- The backplane must be capable of delivering more current (as indicated by the minimum backplane current for each slot type) at the same time. However, the chassis power supply is unlikely to be able to deliver this current to all slots at the same time. The backplane limit imposes a restriction on how much current a module can draw on each supply without risk of damage.
- For requirements on Hybrid Chassis see the section on Hybrid Chassis.

System Slot

The System Slot can be used to host an embedded PC or a remote controller interface (including those based on the External Cable PCIe standard). Note that this controller slot is different to that used in PXI, the two are mechanically and electrically incompatible; a PXIe controller cannot be used in PXI, or a PXI controller in PXIe.

System Timing Slot

This slot is dedicated to supporting the timing functions of PXIe. Unlike PXI it cannot be used for any other purpose (it cannot accept a Peripheral Module), so its inclusion in a chassis can mean one slot is unavailable for any other purpose. This has led to chassis being offered without timing slots, and therefore without the Star Trigger support.

PXIe Modules

As with PXI, PXIe modules in principle can be provided in 3U and 6U form factors and may be arranged to support dual 3U stacked modules. 3U modules have one ejector handle whereas 6U modules have two ejector handles.

PXIe 3U modules connect to the backplane thru the XJ3 connector to provide PCIe and timing control and XJ4 to provide the PXIe instrumentation functions (triggers and clock) and power. On 6U modules an additional optional connector, XJ8, can provide additional power to the module.

As with PXI it is good practice to ensure that all PXIe modules are secured with fixings top and bottom firmly secured and the ejector handle locked.

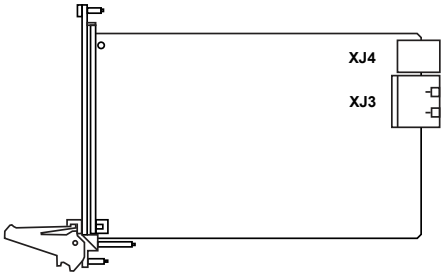


Fig. 2.4 - 3U PXIe Peripheral Module

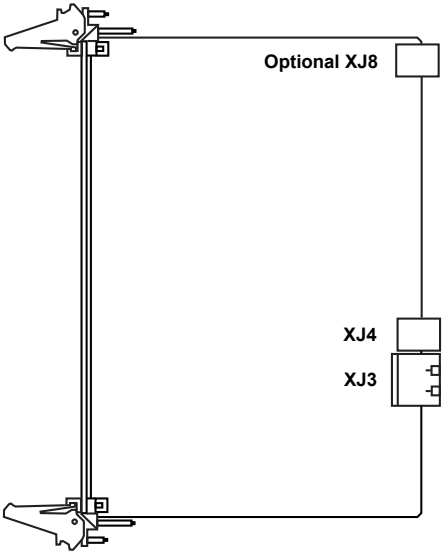


Fig. 2.5 - 6U PXIe Peripheral Module

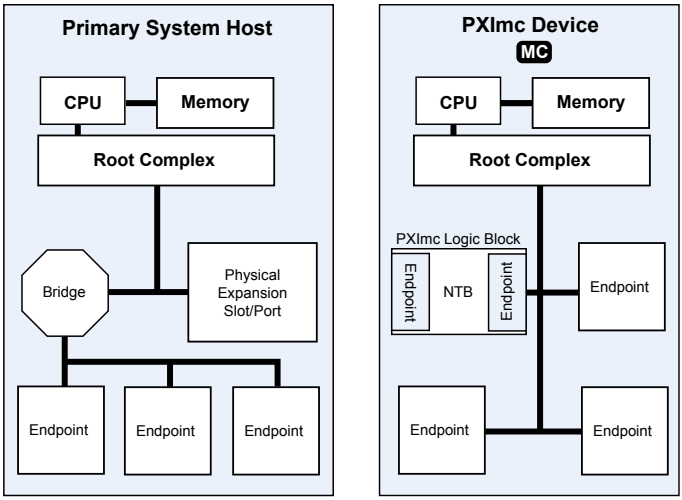
Chassis Recommendation

The issue of compatibility between PXI and PXIe Chassis means that for a user to be sure of being able to use all the available slots, Pickering Interfaces recommends the use of PXI unless the system benefits from having PXIe. In that case we strongly recommend a fully Hybrid Chassis as described later.

PXI MultiComputing (PXImc)

PCIe systems are primarily constructed on the basis of having only one controller, all communication routes between the module and the controller. This arises because the controller is writing and reading into memory over a PCIe interface which requires the interface between the Root Complex and the end point to be synchronized to each other. The Root Complex is the master and the system cannot have two Root Complexes. So to share information between two PCIe systems a different approach is needed.

There is believed to be a demand to distribute the computing requirements in a system, this is the case for example with GPIB/LXI instruments where many of the instruments have their own controller that processes the measurement data and reports the results. Distributed processing can lower the need for high-end fast controllers, and speed a test system up by essentially having multiple computers working on different aspects of a test task rather than a signal central controller having to do all the work.



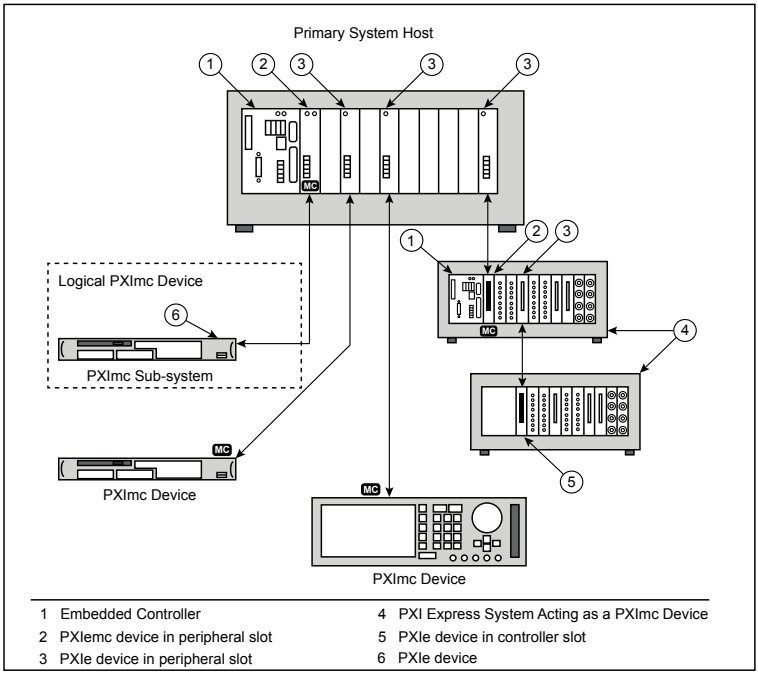


Fig. 2.6 - A PXImc system

To address this requirement PXImc was introduced which allows other controllers to be added to the PXIe system.

> Note: Strictly PXImc can be added to PXI systems, in practice it is only likely to be added to PCIe interfaces.

In PXImc an End Point is created which has a Non Transparent Bridge (NTB) connecting it to another PCI End Point in a different PCIe system. It creates an interface where the two PCIe systems can each lock their clocks and data interfaces their side of the NTB and the NTB systems handles the exchange of information between the two End Points. As the End Points on each side are running from different masters the system is asynchronous and transferring data from one to the other involves some additional latency.

The interface is Non Transparent because neither controller can have visibility of the PCIe bus that lies beyond the PXImc interface, just as in GPIB/LXI instruments where the workings of the instrument are largely not visible to the system controller.

Although at the time of writing (2014) controller speed is a limiting factor in some high end applications, uptake of PXImc has been limited with only National Instruments introducing products.

PXImc is not the only way an NTB system can be implemented on a PCIe bus. There are proprietary NTB's provided by semiconductor vendors and systems integrators.

PCIe External Cabling

PCIe is usually associated with backplane connection systems on PCB's but there is an External PCIe cabling standard created by the PCI-SIG in 2007. This standard allows lanes of PCIe to be exchanged over a cable system. The standard defines the connectors used for different lane counts (x1, x4, x8, x16) but does not define the cable itself – rather is sets the performance the cable must meet.

PXIe and PCIe systems can be expanded using modems that include this type of cable interface. The modems can also support PXImc or other NTB's.

Image credit Molex Inc.

Fig. 2.7 - Examples of PCIe external cabling and the mating connectors for lane widths from 1 to 16.

SECTION 3

Hybrid Chassis

BACKGROUND

As can been seen from the previous sections there are significant differences between PXI and PXIe, and that presents a backward compatibility problem. The number and variety of PXI modules from different vendors means that user choice of modules is constrained if a PXIe only chassis is used, and that is likely to be the case for many years to come. Many applications, including switching, do not need high data bandwidths and the differences in power supply specifications and trigger operations mean that many vendors may choose not to use PXIe. This was anticipated when the PXIe standard was created, so the standard allowed for the combining of PXI and PXIe in the same chassis - a Hybrid Chassis. The chassis has to be designed to support this mode of operation and there are two ways of managing compatibility, the inclusion of Legacy or Hybrid Peripheral Slots.

Legacy Slots

In a Hybrid Chassis a Legacy Slot is slot specifically designed to ONLY support PXI modules. It has the conventional 32/64 bit PCI and PXI compliant power supplies. Only the PCI interface can be used to control the module, and the slot should accept any PXI module. Some features may be restricted (for example Star Trigger, Local Bus) and only a few slots may be supported in the chassis. It is the lowest cost (to the manufacturer) route, but means that users have to be careful about selecting modules to match the available slots.

Hybrid Slots

In Hybrid Slots both the PCI and PCIe interfaces are present. The chassis delivers power compliant to both versions of the standard. Only one interface is permitted to be used at a time, and that has to use the corresponding set of power supplies and triggers (otherwise modules would be created which required Hybrid Slots with both connector sets).

From a user perspective Hybrid Slots in a PXIe Chassis provide the flexibility to use either type of module. PXI modules, which are often not high data users, can be placed in any Hybrid Slot. This has resulted in chassis being offered that are entirely Hybrid with the exception of the System Controller and the Trigger Slot (if available). It tends to be a more expensive solution to manufacture because the chassis has to include both sets of power supplies and both PCI and PCIe interfaces on every slot. The PXI modules can only use one of the Local Bus connections, but it is unusual for PXI modules to use the Local Bus.

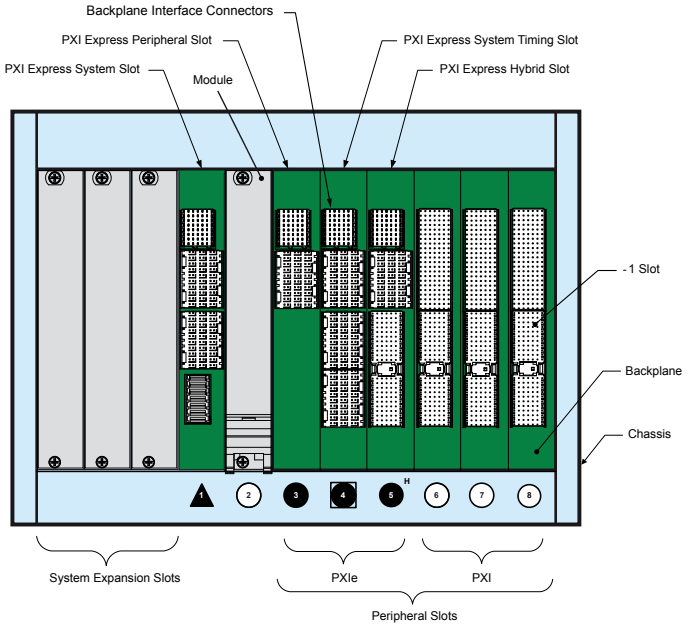


Fig. 3.1 - A PXI Express Chassis showing slot types and a module

Not all PXI modules are compatible with PXIe Hybrid slots, to be compatible the module must not fit the full upper connector – it must have either no upper connector (as cPCI) or the shortened version of the connector. Figure 3.2 shows PXI modules that will fit into a PXIe hybrid slot, while the module in figure 3.3 will only fit in a regular PXI Chassis.

Fig. 3.2 - A PXI Module without a rear upper connector (left) and a short upper connector (right)

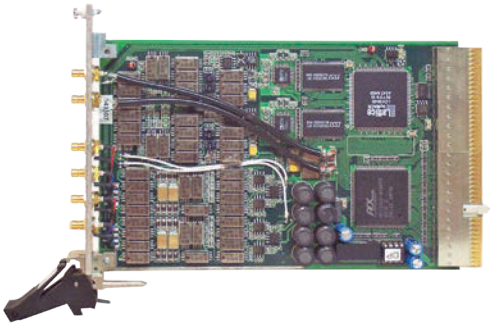


Fig. 3.3 - A PXI Module with a full rear upper connector.

Chassis Power Requirements

A Hybrid Chassis must support the power needs for both PXI and PXIe modules, but it does not need to supply the power to both the PXIe and the PXI connections (in fact it is forbidden, you use either the PXIe control and PXIe power, or PXI control and PXI power). The chassis needs to meet the power requirements set for both PXI and PXIe on a per slot basis (but not at the same time on any individual slot). This of no great concern on 3.3V and 12V supplies where the PXIe Chassis places greater demands on the power supplies, but users need to check the 5V and -12V supply capability, the latter having no PXIe equivalent.

Backplane Connectivity

The controller for a Hybrid Chassis has to be a PXIe controller, not a PXI controller. The PCIe interface from the controller supports the PXI modules by the inclusion of a PCIe to PCI Bridge to supply the PXI modules in Hybrid and Legacy Slots. This clearly creates a little more complexity in the backplane design, and adds more PCI bus segments.

SECTION 4

From Backplane to Module

FROM BACKPLANE TO MODULE

The PCI(e) backplane provides a very fast method of transferring data with low latency at the physical layer. Some of the headline latency and transfer rates need to be treated with a high degree of caution though – they are not often reflected in the actual speed or latency of moving data into the PXI(e) module and for the module to respond. The speeds quoted are often just the physical layer and take no account of software overheads or the hardware issues encountered in the module. The following section highlights some of those issues. PXI(e) does provide a fast IO mechanism, but for most implementations module response time is not limited by the IO speed.

PCI(e) Hardware Interface

Data from the backplane, whether by PCI or PCIe, can arrive at a high data rate. An interface device is used to buffer this data from the backplane to the module and handle all the low level transactions on the PCI(e) interface.

While in principle it is possible to design hardware that can use the data at the rate it arrives such an arrangement has many practical problems – including implementation costs. Most modules will use a serial loop that transfers data from the backplane interface to the hardware within the module and the most common interfaces for this are SPI and I^2C, both of which have transfer speeds significantly below (by several orders of magnitude) the backplane data rate.

There are devices, such as high speed memory/registers, that can handle the backplane data rates more or less directly but these take space and create additional costs, so they tend to be reserved for applications where vast amounts of data need to be transferred from a relatively simple module to the intelligent controller for the processing of the data. In some cases this transfer might be after an FPGA which processes the data in hardware to reduce the burden on the system controller and reduce the amount of data to be transferred. These devices however tend to be the minority of PXI modules, most have much more mundane requirements.

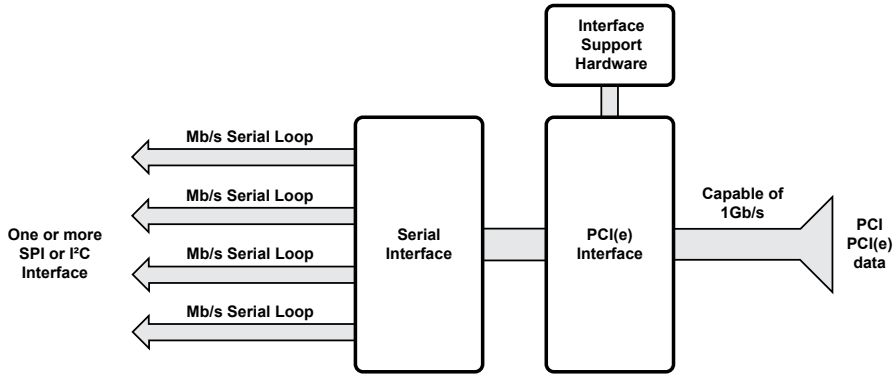


Fig. 4.1 - PCI(e) Interface

For the majority of modules where data transfer rates are not demanding it is likely that module area is better expended on providing room for hardware not related to the data transfer, for example providing more channels on a data acquisition system, or more relays in a switching module.

A common method of transferring data is to use a commercial interface, for example from PLX or IDT, that transfers the backplane data to a set of digital outputs. The interface may have limited amounts of internal buffering to buffer backplane data but it is quite common for this to be very limited or not usable. The driver for the module is then likely to operate by “bit bashing” data serially from the backplane through the SPI or I^2C interface to the module hardware. For the duration of this process the backplane interface will be held up by the serial interface in the module. For example, if a serial interface is running at 3Mbits/s and the serial loop is required to be 300 bits long then the transaction (excluding any other factors) will take 100µs. This method of handling the interface is simple for the module hardware and is a low cost solution. It also minimizes the footprint of the interface and that can be important.

The interface can also be implemented with an FPGA, whereby a core is acquired from the FPGA tool vendor and used to implement the backplane interface, the FPGA can add buffers which buffer the data and allow the data to be clocked out at a more leisurely rate through the SPI or I^2C interface. As long as the buffer is big enough this approach releases the backplane more quickly and ties up the system controller for less time.

When an FPGA is used on PXI it has to be tolerant of 5V backplane signalling. As FPGA designs have migrated to smaller geometry the 5V tolerance has become less available and interface devices are required to protect the FPGA while conforming to the PCI specification.

Use of FPGA designs does add an additional burden on PXI vendors to ensure their designs robustly comply with the PCI(e) specification so this technique tends to be less commonly used by smaller vendors or vendors providing the lowest cost solutions.

Whichever interface method is used the IO speeds are very high, but they are not often comparable with the headline speeds associated with PCI(e) interfaces.

Controller

Another aspect of PXI(e) systems that users need to understand is the standard focuses on using the system controller as the intelligence for all of the modules. In some cases tasks may be off-loaded to module hardware (for example an FPGA), but for most modules the task of interpreting an instruction from the application program (for example an instruction to close a specific relay) and converting it into data controlling hardware falls to the driver on the system controller. The system controller is assumed to be very fast, but the faster the controller the more expensive it is likely to be and its resources are spread across all the modules in the system and to running the application program. For relatively simple tasks controller overheads are not high so the driver will complete its task quickly, but not as quickly as the headline numbers on bus latency.

There is also likely to be an overhead associated with the controller because of its operating system. Many systems use a Windows environment, some use a Linux environment, and others may use a Real Time Operating System (RTOS). In all cases latency is introduced by the operating system which is far greater than the latency of the backplane interface. Windows in particular is well known for diversions to housekeeping tasks that can last for 100's of milliseconds on some occasions. Real time operating systems are not necessarily about low latency, they are primarily about determinism – ensuring a task is prioritized and completed within a certain time. That time could be short or it could be relatively long, too many short times can simply overload the controller so tasks need to be prioritized.

Switching System Latency

The primary product provided by Pickering Interfaces is switching systems. The product range includes three primary types of relays:

- Reed relays which offer operating times measured in 100's of microseconds.
- Electro-Mechanical Relays (EMR's) which offer operating times measured in a few milliseconds to 10's of milliseconds.
- Solid state relays which offer operating times measured in a few microseconds to 100's of microseconds.

For these products IO speed and controller latency tend to be just a small part of the total operating time with the possible exception of some solid state designs where the switch operating speed is much faster.

A key function provided by the driver program on the controller is to manage timings so that a user has no danger of accidental transitory paths as switches are first opened and others are then closed. In some cases there are several relay operations required to change a switching system state which, depending on the switching system, may look something like this:

- Open relays that are closed and need to be set to open.
- Change the setting of any isolation relays that are present that optimize parameters such as hot switching capacity or system bandwidth.
- Close relays that are presently open and need to be set to closed.

Much of this can be handled in the driver when IVI is being used. The ideal sequence will require delays between the events to ensure that relays have operated before the next action is taken, ensuring no unintended transient paths and that isolation relays do not get worn out by hot switching operations. So in reality some switching architectures require several operations to complete a system change, and that can amount to a very long time compared to the IO speeds.

Switching systems are not the only module functions that have this time restraint. Products such as DMM's in reality need to integrate measurements over time to reject

noise – and if that noise involves rejection of AC supply signals then speeds are limited to integrals of the AC supply frequency. A user can of course elect to do this by averaging in the controller rather than the DMM, but that simply imposes a task on the resources of the controller instead of the hardware of the DMM.

Switching System Timing Examples

Different PXI(e) vendors will implement their interfaces in different ways. This is driven by the need to meet market demand on costs, and timings may well have a dependence on the power of the controller and how busy it is with other tasks (including for example operating system tasks). The following example may give an indication of what might happen in a reasonably dense PXI(e) switching module:

A Switching System consists of 256 relays on a single serial loop connected to the backplane by a commercial PCI(e) chip set. The serial loop transfers data at 2 million instruction cycles per second.

Two examples are shown, in the first a single operation is used to set a relay state, this is the value most commonly quoted on data sheets from switch system vendors.

In the second example two operations are performed, the first to open all relays that may require to be set to be open, the second to close any relays that require closing. This sequence is used when there are risks associated with creating inadvertent connections, particularly when power supplies or sensitive signal lines are being connected.

Timing Sequence for Single Operation

Event	Time
Process the switching system request from the API and open the module.	50µs
Load new settings into the serial loop	128µs
Wait for relays to settle, assume EMR at 3ms	3000µs
Operation complete, total time	**3178µs**

Timing Sequence for Sequenced Event

Event	Time
Process the switching system request from the API and open the module.	50µs
Load new settings into the serial loop to open any relays that require opening	128µs
Wait for relays to settle, assume EMR at 3ms	3000µs
Load new settings into the serial loop to open any relays that require closing. Requires 256 instructions and transfer to outputs that set the relay coils	128µs
Wait for relays to settle, assume EMR at 3ms	3000µs
Operation complete, total time	**6306µs**

During this time interval the system controller is not necessarily tied up with operating the switching system, it can also be working on other tasks. The PCI bus may also be available to control other modules.

For switching systems it can be seen that operating times are dominated by the relays, in this case EMR's. If reed relays or solid state switches were used the times would reduce substantially – for reed relays the times are similar to the interface overheads.

Summary

PXI(e) systems do offer very high speed IO, but users should not expect these high IO speeds to directly translate into radically faster test system operating times. In many cases test system speed is limited by more practical issues in the system. It can be faster than alternative approaches and vendors can always choose use cases carefully where this is the case. In most systems the advantages of PXI are more about its ability to integrate multi-vendor solutions into a chassis capable of supporting a diverse range of products leading to a small footprint solution. As long as users are clear on this and they make careful selection of the required PXI Modules, they will not be disappointed with PXI(e) solutions**.**

SECTION 5

Software

INTRODUCTION

The PXI standard is reliant on a standardized software and hardware environment. PXI modules have no front panel controls and rely entirely on software control via the PXI backplane.

PXI modules appear on the PCI bus of the controller and so installation of a PXI module is almost identical to that of a PCI card.

Operating Systems Supported

The PXI standard requires that PXI modules must support 32-bit Windows or 64-bit Windows, commonly both are supported.

It can be assumed that versions of Windows supported by Microsoft will be supported, although there may be a lag between the release of a new Windows version and the availability of drivers.

At the time of writing most vendors will provide support for:

- Windows XP
- Windows 7
- Windows 8

Support for earlier versions of Windows may also be available, but since these are no longer fully maintained and supported by Microsoft, it cannot be assumed they will be provided. Note that support for Windows XP ceased in April 2014, and Windows Vista support ended in 2011.

As operating systems evolve, some compatibility problems may occur. For example Windows 8 requires signed drivers, whereas Windows XP did not, so a driver developed for Windows XP may not install on Windows 8. Always check with the hardware vendor that the operating system to be used is fully supported.

Also take into account that most 32-bit drivers will work on a 64-bit system, the use of 64-bit Windows does not necessarily dictate the use of 64-bit drivers.

Other Operating Systems

Other operating systems may be supported but this is not a requirement of the PXI standard. If the user is planning to use any other system, checks must be made with the hardware vendors for availability of software support.

To successfully operate a PXI platform the operating system must be able to connect to the PXI bus and driver software must be available to support that operating system.

LabVIEW Real Time

LabVIEW Real Time requires the use of a VISA driver and therefore most PXI modules should operate correctly using the Windows driver. However, check with the vendor to make sure.

Installation to the LabVIEW Real Time system may involve transferring a number of files to the LabVIEW Real Time target system using either the ftp tool provided with LabVIEW Real Time or almost any ftp client application.

Linux

Linux is increasingly being adopted. However, unlike Windows it is not possible to provide a single driver that will work on any system. The driver must be specifically compiled for the Linux kernel in use. Check with the PXI card vendor for support on the particular Linux system being used, in general the vendor will need to know the precise Linux distribution being used. Some versions of Linux, particularly real-time versions, are not generally available and may present problems to the PXI card vendor.

Development Environments Supported

The PXI specification also recommends that a number of Development Environments be supported:

LabVIEW	(National Instruments)
LabWindows/CVI	(National Instruments)
ATEasy	(Geotest-Marvin Test Systems Inc.)
Visual Basic	(Microsoft)
Visual C/C++	(Microsoft)

However, none of these are mandatory, so check with the vendor. Most vendors also provide support for Visual C# (Microsoft).

Register Level Interface

Where no driver is available for the OS chosen for the test system, it may be possible to control a PXI card using low-level register level control. This approach requires that the programmer has detailed information of the hardware and control techniques therefore can only be considered if the PXI module vendor is willing to provide this level of detail.

This route to module control is not recommended except in exceptional circumstances, it is likely to require a great deal of assistance from the module vendor.

Check with the vendor before embarking on this approach.

DRIVER MODEL

On most operating systems, including Windows, the user cannot interact directly with the hardware but must access through a driver designed for the purpose.

The kernel driver provides the low level hardware access in kernel space and exposes an interface in user space. The kernel driver provides only a very basic low level interface and typically a further module Application Programming Interface (API) builds on the kernel to provide an interface better adapted to the control of that particular module.

More advanced APIs may build on the lower levels to provide increasingly useful interfaces to include further features and enhancements.

An application program may access the hardware module using any of the available APIs, choice will depend on a number of factors such as the programming environment, interchangeability requirements, and even personal choice.

This diagram shows a typical set of choices available, from low level programming using the kernel driver interface through increasing higher level APIs that provide progressively better modelling of the functionality of the particular hardware module.

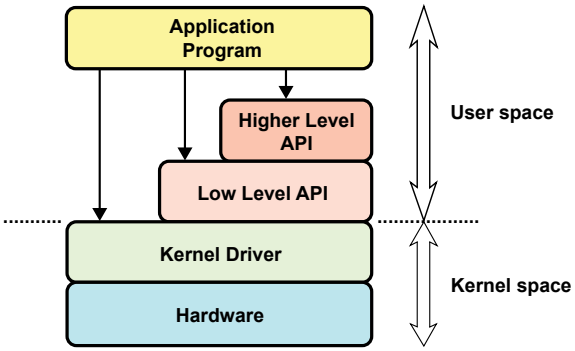


Fig. 5.1 - Programming layers.

VISA is a kernel driver providing hardware control plus resource management. The interface is low level, providing only basic input/output functionality for module control. Module control at this level may be very complex and require detailed understanding of the card hardware. Almost all manufacturers will provide a low-level API that encapsulates specialist knowledge of the hardware module to simplify the programming task.

An IVI driver is provided by many manufacturers. This is a higher level API which builds on the lower driver and may conform to industry standard functionality for the module type.

There may also be more layers than shown above.

In many cases a non-VISA set of drivers will be available. This is useful in cases where VISA is not available, either due to operating system or licensing limitations. For example VISA is only available on a limited number of Linux distributions so the user will be forced to use an alternate kernel interface.

CHOICE OF DRIVER

A VISA interface driver is required by the PXI standard, however many PXI modules are provided with a selection of drivers. The user must select the driver most suited to their application and that may also involve an element of personal choice.

Increasingly IVI (Interchangeable Virtual Instrument) drivers are provided. This driver standard is aimed specifically at interchangeability which is discussed in a later section. It may also be required for particular software tools, notably Switch Executive from National Instruments which will only handle modules with an IVI Switch class driver.
In some cases a user may elect to construct a system without using VISA. In this case it is essential to consult the hardware vendors to verify if a suitable driver is available.

VISA

The VISA standard - Virtual Instrument Software Architecture – was originally created by the VXI plug & play system alliance and is now maintained by the IVI Foundation (www.ivifoundation.org).

The objective of the standard is to define a way of creating instrument drivers with a degree of interoperability between different manufacturers' modules.

The PXI standard encourages the use of the VISA standard.

Key aspects of VISA are:

- It allows you to install different drivers from different manufacturers on the same PXI system without conflicts.
- It uses a standardized VISA I/O layer for all I/O functions to ensure interoperability.
- It defines a way of writing drivers.
- A driver that follows the VISA specification uses defined data types and in some cases defined function names.
- It reduces the process of learning new instruments and the time to develop a test system.

IVI

The IVI standard (Interchangeable Virtual Instrument) is supported by the IVI Foundation (www.ivifoundation.org). The aim of IVI is to give a degree of interchangeability, instrument simulation and in some cases, higher performance. IVI supports all major platforms including PXI, AXIe and GPIB. Being a higher level interface which often uses

a lower level driver for the hardware interface, its use may result in slightly slower speed compared to other drivers.

Goals

The stated objectives of the IVI Foundation are to improve hardware interchangeability by:

- Simplifying the task of replacing an instrument with a similar instrument
- Preserve application software if instruments become obsolete
- Simplify code re-use from design validation to production

Improve quality by:

- Establishing guidelines for driver testing and verification

Improve interoperability by:

- Providing an architectural framework that allows users to easily integrate software from multiple vendors
- Providing a standard access to driver capabilities such as range checking and state caching
- Simulating instruments to allow software development when hardware is not available
- Providing consistent instrument control in popular programming environments

As with VISA, IVI is a way to standardize driver development but it goes much further.

The set of IVI specifications provides a number of instrument class definitions, each class has a standard interface for programming, including function names and data types. By appropriate use of IVI class drivers a user can develop a system that is hardware independent, meaning instruments may be easily changed for similar instruments from different vendors without the need to re-code the users application.

At the time of writing the following classes are defined:

IVI-4.1: IviScope Class Specification
This specification defines the IVI class for oscilloscopes.

IVI-4.2: IviDmm Class Specification
This specification defines the IVI class for digital multimeters.

IVI-4.3: IviFgen Class Specification
This specification defines the IVI class for function generators.

IVI-4.4: IviDCPwr Class Specification
This specification defines the IVI class for DC power supplies.

IVI-4.5: IviACPwr Class Specification
This specification defines the IVI class for AC power sources.

IVI-4.6: IviSwtch Class Specification
This specification defines the IVI class for switches.

IVI-4.7: IviPwrMeter Class Specification
This specification defines the IVI class for RF power meters.

IVI-4.8: IviSpecAn Class Specification
This specification defines the IVI class for spectrum analyzers.

IVI-4.10: IviRFSigGen Class Specification
This specification defines the IVI class for RF signal generators.

IVI-4.12: IviCounter Class Specification
This specification defines the IVI class for counter timers.

IVI-4.13: IviDownconverter Class Specification
This specification defines the IVI class for frequency downconverters.

IVI-4.14: IviUpconverter Class Specification
This specification defines the IVI class for frequency upconverters.

IVI-4.15: IviDigitizer Class Specification
This specification defines the IVI class for frequency digitizers.

It is important to remember that the class definition cannot include any vendor-specific features, it contains only the basic functionality of the instrument type. It also cannot take into account differences in performance, such as accuracy or speed. In practice it is essential that consideration be given to the consequences of changing from one manufacturer's module to another since those modules may not behave in exactly the same way.

IVI drivers have built-in simulation capability. With this simulation feature is it possible to develop an application without the instrument being present which means software development may start before instruments are delivered, or while being used in another application.

IVI Driver Architecture

An IVI Driver is a driver that implements the inherent capabilities defined in the IVI-3.2 Inherent Capabilities Specification document, regardless of whether it complies with a class specification.

An IVI Class Driver is a generic abstract class defining the basic features of instruments of that class as agreed by the IVI Foundation members. An IVI Specific Driver contains features specific to a particular vendor which may not be applicable to modules from other vendors. IVI Specific Drivers may be further sub-defined as an IVI Class-Compliant Specific Driver or as an IVI Custom Specific Driver. An IVI Class-Compliant Specific Driver provides both the Class functionality and additional vendor-specific functionality.

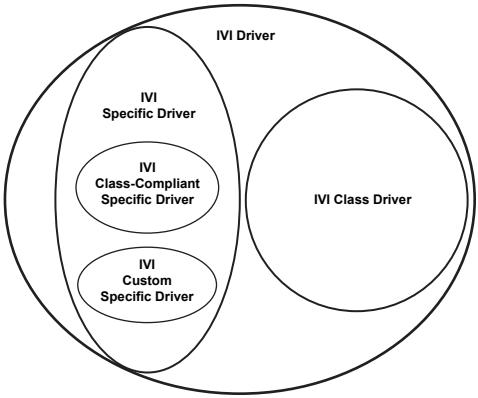


Fig. 5.2 - The IVI Driver.
(Reproduced from the IVI-3.1 Specification)

Most specifications contain optional class extension capabilities, such as the Scanner function group in IviSwtch. Being optional it cannot be assumed that all vendors will provide these capabilities.

Most IVI drivers fall into the IVI Class-Compliant Specific Driver group. This means that the driver is class compliant but adds further functionality beyond the class definition.

IVI Class Driver	IVI Class Compliant Specific Driver	IVI Custom Specific Driver
Inherent Capabilities	*Inherent Capabilities*	*Inherent Capabilities*
Base Class Capabilities	*Base Class Capabilities*	
Class Extension Capabilities	*Class Extension Capabilities*	
	Instrument Specific Capabilities Functions defined by the manufacturer	*Instrument Specific Capabilities* Functions defined by the manufacturer

Further to the above, drivers may be provided with a C interface, COM interface or .NET interface.

Most development environments are capable of interfacing to a C interface driver, and many to a COM interface; whereas the .NET interface has a more restricted range of environments.

The IVI Configuration Store

Central to the IVI Driver model is the IVI Configuration Store. This diagram shows the relationship between the various software elements involved when using the IVI system.

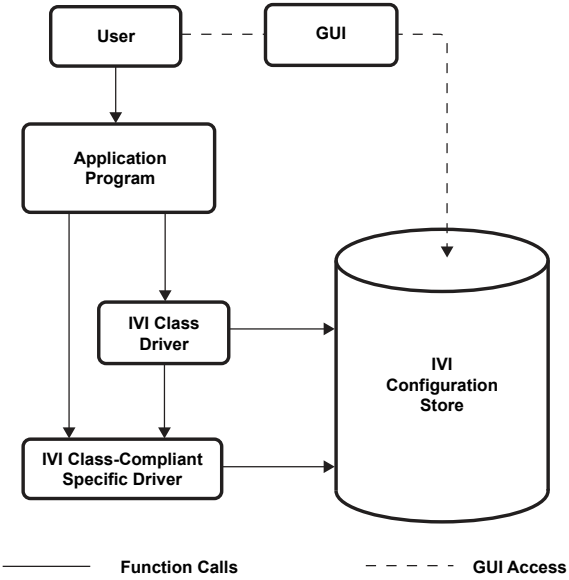


Fig. 5.3 - Using the IVI-C Class Compliant Specific Driver.
(Reproduced from the IVI-3.1 Specification)

The IVI Configuration Store is an XML file containing definitions and relationships between the various aspects of a module and its software driver. The IVI software system provides means to access the store from a driver. Tools to access and manipulate the IVI Configuration Store are available, notably National Instruments Measurement and Automation Explorer (MAX).

EXAMPLE OF INTERCHANGEABLE SWITCH MODULES

The IVI Switch Class driver is the key to interchangeability, using this driver allows differences between software implementations from different vendors to be moved from the user application and dealt with by the IVI software system.

In the example shown in Figure 5.4 a pair of changeover relays are used to connect one of two devices under test (DUT) to a signal generator and a spectrum analyzer. For this application a coaxial RF switch is required, National Instruments' PXI-2599 and Pickering's 40-780-022 are both suitable for this application, however they use different drivers and have different naming conventions for the switch channel names as shown in the diagram.

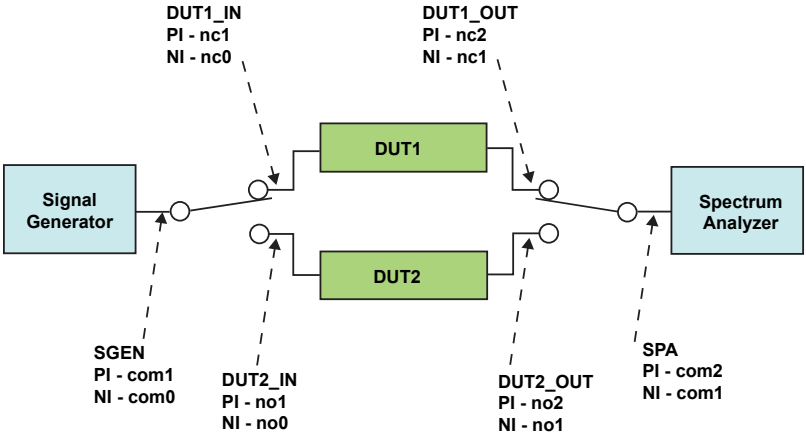
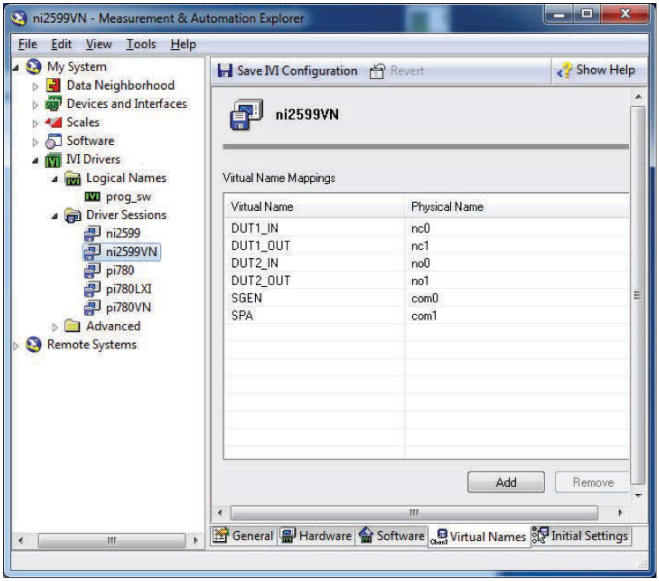


Fig. 5.4 - Changeover relays connecting devices under test

The first step toward interchangeability is to define virtual names for the channel names; these virtual names will be employed in the user application. Figure 5.5 provides screen shots from NI MAX showing the Virtual Name tables for the NI-2599 and the Pickering 40-780-022 where the differing naming conventions of the two cards are mapped to a common set of Virtual Names.



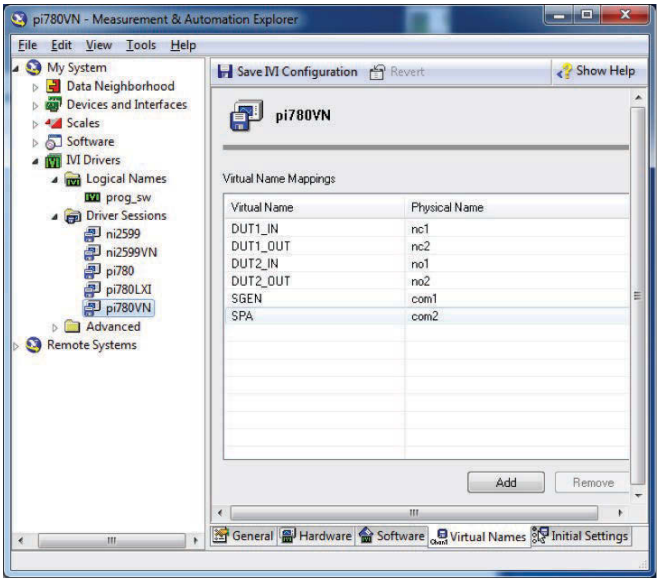


Fig. 5.5 - Defining Virtual Names

Next a level of indirection is employed to decouple the specific drivers from National Instruments and Pickering Interfaces from the user application. The IVI Configuration Store provides this indirection; it creates the concept of a Logical Name which 'points' to a Driver Session, this linkage may be changed within the store such that a Logical Name may be altered to refer to a different Driver Session. So, if all the differences between the NI and Pickering drivers can be encapsulated in a pair of Driver Sessions, then the Logical Name can be simply modified to refer to either Driver Session. The user then creates an application using the Logical Name. If at some time the alternate module is to be used, then the Logical Name may be changed to refer to the alternate Driver Session. So, the switch module may be replaced with one from a different vendor just by changing the linkage of the Logical Name.

In the screen-shot from MAX shown in Figure 5.6, the logical name 'prog_sw' is linked to the driver session for the Pickering 40-780-022.

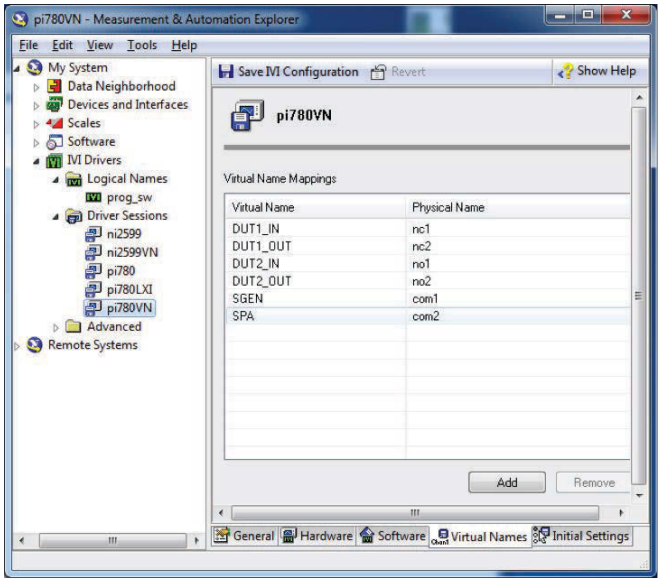


Fig. 5.6 - A screen-shot from NI MAX

It must be remembered at all times that only differences in software implementations can be interchanged, hardware and performance differences cannot be.

The user application should code using the IVI Swtch Class Driver thus:

```
err = IviSwtch_init(“prog_sw”, 0, 0, &vi);
err = IviSwtch_Connect(vi, “DUT1_IN”, “SGEN”);
err = IviSwtch_Connect(vi, “DUT1_OUT”, “SPA”);
```

This code uses a Logical Name to identify the hardware/software combination and uses Virtual Names to identify the switch terminal channels. It provides completely interchangeable code in that the Logical Name and the Virtual Names may be manipulated in the IVI Configuration Store at any time to permit this code segment to operate different switch modules from different manufacturers without the need to modify the code.

If at some time in the future a new switch product from a different vendor becomes available, all that is required is to create a new Driver Session that defines the driver to be used and the Virtual Name table to define the relationship to the channel names exported by that new driver. The Logical Name may then be modified to link to the new Driver Session and the user application will then use the new module without the need to modify or re-build the application.

SECTION 6

LXI, USB in PXI

Introduction

The PXI specification leaves much room for innovation in PXI and PXIe. In this chapter, we will show two ways where Pickering interfaces has added new ways to use PXI in applications.

PXI IN LXI ENVIRONMENT

In PXI systems the PXI modules appear as extensions of the controller's PCI internal system, it is as if the modules were mounted inside the PC.

This is a good arrangement for most applications, but sometimes users want the functionality of PXI modules in an environment that is less sensitive to controller updates and has easier connection at a distance. The LXI standard is based upon Ethernet as its control interface and addresses many of these applications.

Ethernet physical layer latency is significantly higher than PCI (the Ethernet Alliance indicating 16us typical on a 1000BaseT interface) and also has latency introduced by the controller that processes the user instructions. Typically an Ethernet interface is less powerful than a PXI system controller (Pickering Interfaces' LXI solutions currently use an ARM based controller running Linux) so it takes longer to execute commands, but it does off-load tasks from the system controller so it creates a degree of distributed control.

PCI(e) is a commonly used internal interface used in many instruments so many embedded controllers have a PCI(e) interface that can be used to control PXI modules.

Pickering Interfaces' 60-102 and 60-103 series Modular Chassis are examples of LXI modular chassis that can be used to control Pickering Interfaces' PXI modules.

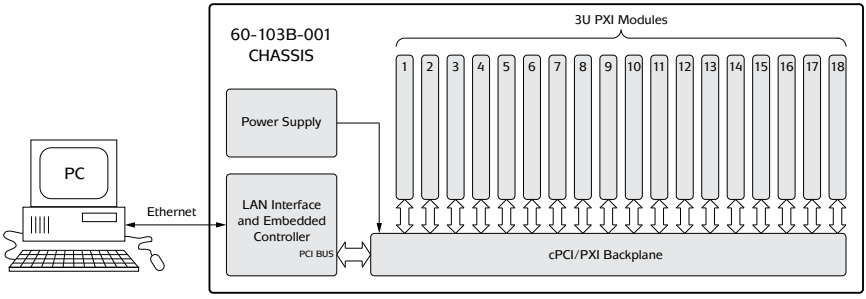


Fig. 6.1 - Plugging Pickering Interfaces PXI modules into the 60-103B LXI Modular Switching Chassis provides an LXI compliant switching platform.
(A simple Ethernet lead is the only physical control connection required.)

In the 60-103B a PXI Chassis is adapted so it has a an LXI compliant system controller. It has access to the control features of the PXI Chassis (fan speed, temperature monitors)

that can be accessed over a web interface. It also has access to the PXI modules fitted in the 18 Peripheral Slots that are available via the controller's PCI interface. With the software loaded into the controller it has programmatic control of the PXI modules using almost identical commands as controlling the PXI modules over a controllers PCI interface – the difference is simple editorial change.

This allows the PXI modules to be managed over Ethernet, the internal web server allowing the chassis to be managed over web pages and the soft front panels to be served up from the web interface using Java – an environment available on most controllers without the need for proprietary software.

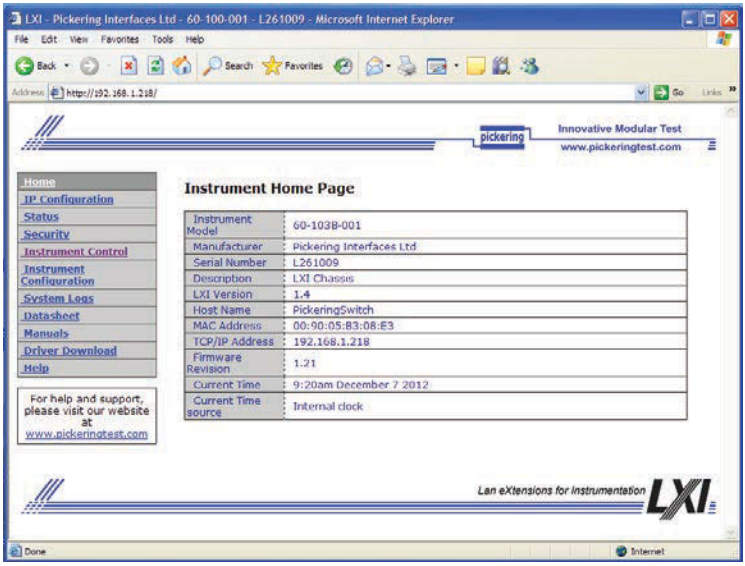


Fig. 6.2 - The home page for the 60-103B provides fast access to configuration information for the LXI Chassis through any web browser.

Supporting PXI modules requires access to source code that needs to be added into the LXI controller. This limits the PXI modules that can be installed into the LXI chassis to Pickering Interfaces products - third party products cannot be supported at the time this book was written.

Pickering's 60-102 and 60-103 modular chassis expand the application areas available for PXI modules, allowing them to address opportunities for switching systems and simulators that would otherwise be more difficult to support.

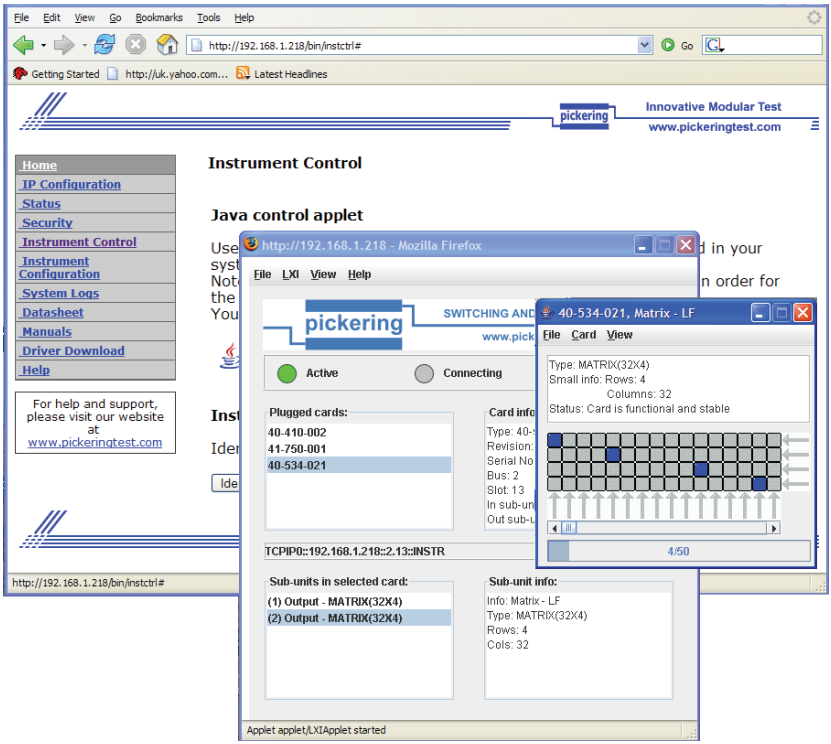


Fig. 6.3 - The 60-103B Soft Front Panels enable manual operation through the web interface, in this example Pickering's 40-534 PXI matrix module
(At the back is the Instrument Control web page; the middle window lists the modules installed with the control panel for the selected module at the front.)

USB

USB instrumentation has gained in popularity for applications where the primary interface is Ethernet (LXI) and a secondary interface is USB or for supporting remote hardware such as power meter detectors. It has also proven popular for applications such as data acquisition where the low interface cost provides opportunities for more cost effective solutions than PXI.

USB connections from a controller are provided from a PCI(e) interface, the same type of interface that controls PXI(e) and again the USB port appears as an extension to the PCI(e) bus. A logical extension of this is to support USB products directly from the PCI backplane present in the PXI chassis. The Pickering Interfaces' USB 2.0 hub (model 40-738) provides a solution for this by providing a USB 8-port hub in a 3U PXI form factor.

In use, the 40-738 allows up to eight devices to be connected to the controller via the PCI bus. Each USB output port of the 40-738 appears just as it would for any other USB device attached to the controller, when connected the USB device can install using the USB vendor driver. Each port provides USB power derived from the PXI backplane rather than relying on the controller power source or an external mains powered hub.

Fig. 6.4 - Pickering Interfaces' 40-738 USB 2.0 Hub

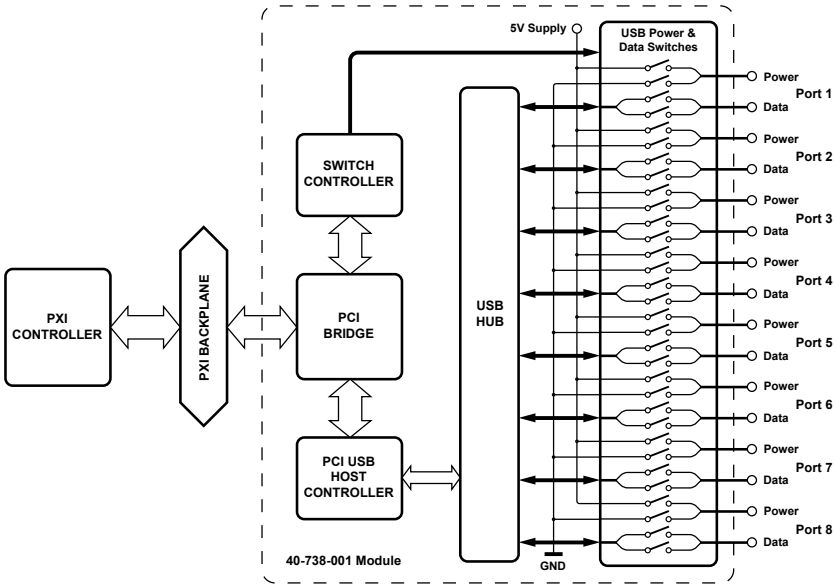


Fig. 6.5 - Functional diagram for the 40-738 USB 2.0 Hub

In addition to providing the USB connection the 40-738 provides a means of switching any of the four data and power lines in each USB port. This allows USB devices to be connected or disconnected to the USB port as though they had been mechanically unplugged. The switch control can be used to re-initialise USB devices.

Some applications have devices under test that have USB ports, the ability to switch any of the four USB lines means fault response and recovery can be checked for.

The switch controls for the 40-738 USB output ports appear as a separate PCI bus so the switches are controlled by the Pickering switch drivers over a non USB interface.

The 40-738 allows USB instrument and test devices to be added to a PXI system with all the parts controlled via the chassis.

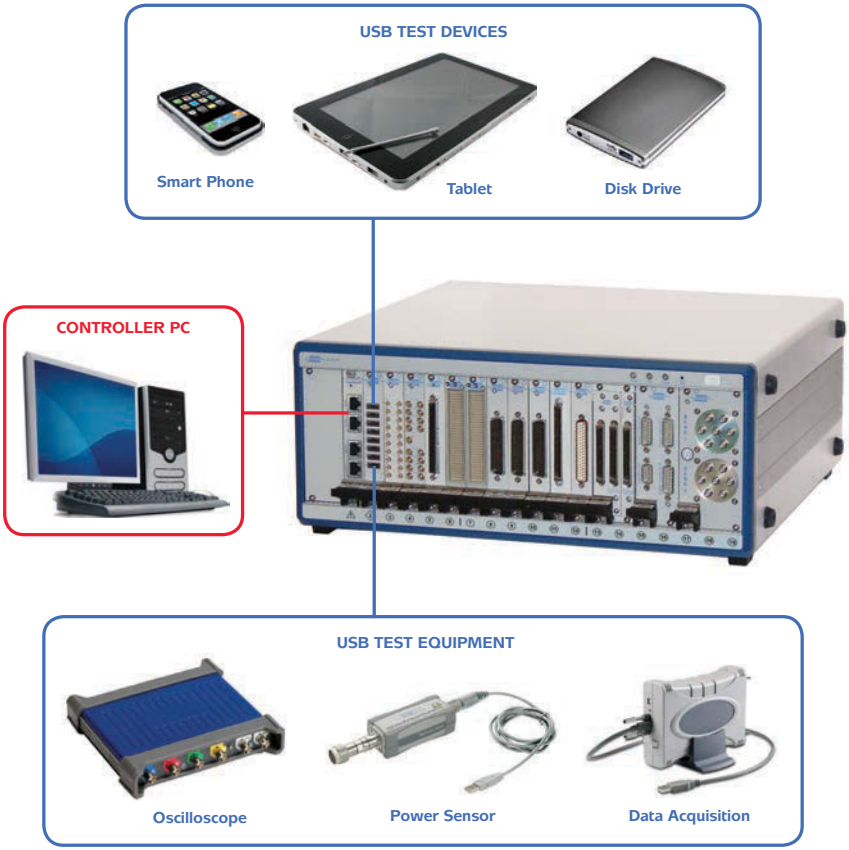


Fig. 6.6 - Expanding a PXI Test System with low cost USB Test Instruments and USB Test Devices

SECTION 7

Pickering PXI Product Overview

ABOUT PICKERING

Pickering has over 45 years of experience in switching technology. Pickering began reed relay manufacturing in 1968 and since 1988 **Pickering Interfaces** has been designing and developing commercial and custom switching and instrumentation products. We maintain a strong commitment to product support, typically 20 years from product launch. Pickering has developed a strong customer base in a wide of range of industries requiring the use of functional test systems.

What we do in PXI

- General Purpose Relay Modules
- Matrix Modules
- Multiplexer Modules
- RF & Microwave Modules
- Fault Insertion Modules
- Programmable Resistor Modules
- Digital I/O, Prototyping & Switch Simulator Modules
- Waveform Generation Modules
- Amplifier & Attenuator Modules
- USB, Communications & Avionics Switching Modules
- Power Supplies & Battery Simulators
- Chassis
- Controllers

For applications requiring diverse switching functions but preferring an LXI interface, Pickering offers an LXI modular chassis. These LXI chassis are capable of hosting Pickering's extensive range of 3U PXI switching and test & measurement modules in an LXI environment, allowing remote control over a gigabit Ethernet connection.

Pickering Relays

A switching system can only be as good as the relays it uses. Reed Relays are hermetically sealed devices with excellent switching characteristics and long mechanical life. Pickering Interfaces works closely with its sister company, **Pickering Electronics**, to design-in the highest quality reed relays available using the very best materials. Our instrument grade reed relays offer a factor of 10 better mechanical life than electro-mechanical relays, with faster operation and more consistent low level operation. Pickering Electronics is the only reed relay manufacturer to utilize ***SoftCenter***® in the construction of its reed relays. For more information on our reed relays go to **pickeringrelay.com**.

Selected PXI Switching & Instrumentation

The next few pages show a selection of our large range of PXI products. To see our entire range of over 1000 switching and instrumentation modules please visit **pickeringtest.com**

General Purpose Relay

Power Relay
(40-160)

SPST High Voltage Power Relay
(40-330)

Reed Relay
(40-110)

Matrix Modules

2 Amp 1-Pole Matrix
(40-505)

10A Solid State Matrix
(40-553)

3U PXI Multi-Slot Matrix
(40-560A)

Multiplexers

5A Power EMR Multiplexer
(40-651)

High Density 10A Power Multiplexer
(40-661)

Power MUX BRIC™
(40-571)

RF & Microwave

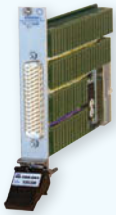
50Ω SPDT RF Switch
(40-870)

50Ω Terminated 6GHz Multiplexer
(40-883)

Microwave Multiplexer
(40-784A)

Fault Insertion

30A Fault Insertion Switch
(40-191)



5A Fault Insertion Switch
(40-198)

BRIC™ High-Density FIBO Matrix
(40-592A)

Programmable Resistors

Load Resistor
(40-292)

High Density Precision Resistor
(40-297)

RTD Simulator
(40-262)

Digital I/O, Prototyping & Switch Simulators

32 Channel Digital I/O
(40-412)

Breadboard
(40-220A)

Dual 16 Channel Automotive Switch Simulator (40-485)

Waveform Generation, Amplifier & Attenuators

Function Generator
(41-620)

6GHz Triple Solid State Attenuator
(41-182)

High Voltage Programmable Amplifier (41-650)

USB, Communications & Avionics Switching

MEMS Fiber Optic Switch
(40-855)

Data Communications MUX
(40-736)

USB 2.0 Hub with Programmable Connect/Disconnect
(40-738)

Power Supplies and Battery Simulators

Dual Programmable +10V Power Supply
(41-735)

Programmable Power Supply 48V
(41-743)

6 Channel Battery Simulator Module
(41-752)

Chassis

8 Slot PXI Mainframe
(40-922)

19 Slot PXI Mainframe
(40-923)

Controllers

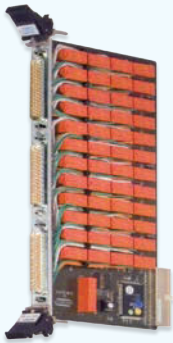
PCI to PXI Remote Control Interface
(41-921A / 51-921A)

PCIe to PXI Remote Control Interface
(41-924 / 51-924)

6U PXI Switching

Ultra High Density Matrix
(45-541)

Expandable 250MHz RF 16X16 Matrix
(45-720A)

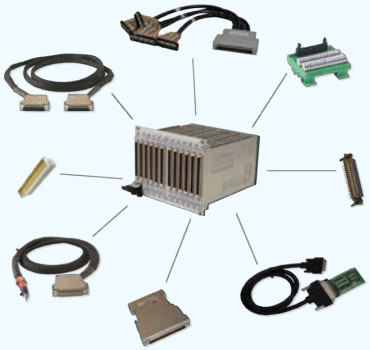


48 x SPDT Power Relay
(45-157)

SUPPORT

Cables and Connectors

Pickering also provides a full range of supporting cables and connector solutions for all of our PXI modules. We offer everything from simple mating connectors to complex cables assemblies and terminal blocks, we also manufacture cable assemblies to special requirements. Every accessory is designed to ensure you have a trouble free experience in connecting our products to your test system.



Mass Interconnect

Pickering recommends the use of a mass interconnect solution when an Interchangeable Test Adapter (ITA) is required to be used with a PXI based test system. The range of Pickering PXI modules are fully supported by both MacPanel and VPC mass interconnect solutions:

MAC-PANEL Interconnect Solutions & Service Since 1958

www.macpanel.com

VPC Virginia Panel Corporation

www.vpc.com

Software

Pickering products are compatible with popular software: Windows® 8/7/Vista/XP, Visual Studio® (VB.NET, C#, C/C++), LabVIEW™, LabVIEW RT™, LabWindows/CVI™, VISA (NI and Agilent), IVI, NI SE, Agilent VEE, Mathworks Matlab, Geotest ATEasy, MTQ Testsolutions Tecap.

BIRST Built-In Relay Self-Test

Selected high density matrix modules are supplied with Built-In Relay Self-Test (*BIRST*). This allows the easy detection of faulty or deteriorating relay contacts to help in the diagnosis and verification of complex switching systems.

SECTION 8

USEFUL INFORMATION

This section provides useful information sources for PXI products and associated organizations and standards.

PXISA

The PXI Systems Alliance is a not for profit organization run by companies involved in PXI products.

The organization is responsible for the production of the technical specifications for PXI and the promotion of the PXI concept in the market.

Membership of the PXISA is open to any company that is willing to promote the PXI standard.

MEMBERSHIP CATEGORIES

PXISA

There are three levels of Membership in the PXI Systems Alliance.

Membership of the PXISA allows participation in the PXISA technical and marketing programs and the use of the PXI logo.

Only PXISA members are permitted to use the PXI logo on their products and marketing material. Executive and Sponsor members must also manufacture PXI products and/or provide services.

Sponsor Member

Sponsor Members, such as Pickering Interfaces, view PXI as a vital part of their ongoing business and want to be involved in all decisions. To be a Sponsor Member the company must be an Executive Member for at least one year prior to applying for Sponsor Membership and be producing PXI products. Sponsor members have a seat on the Board of Directors and voting rights.

Executive Member

Executive Members view the PXI architecture as important to their business. They have a voice in technical specification and approval and Marketing direction of the PXISA. An executive member must also be producing PXI related products. Executive members have voting rights.

Associate Member

Associate Members want to be a member of the Alliance supporting PXI and to be informed of progress. They do not have voting rights and this membership level is for information purposes only.

A membership list and membership application forms are available on the PXISA web site www.pxisa.org

OTHER ORGANIZATIONS

AXIe Consortium

The AXIe Consortium is responsible for the development and promotion of the open AdvancedTCA Extensions for Instrumentation and Test (AXIe) standard.

IVI Foundation

The IVI Foundation promotes specifications for programming test instruments that simplify interchangeability, provide better performance, and reduce the cost of program development and maintenance.

LXI Consortium

The LXI Consortium is responsible for the creation and promotion of the LXI standard. It has a similar structure to the PXISA since it has different membership levels and working groups to create and maintain the specification.

PICMG

PICMG (PCI Industrial Computer Manufacturers Group) is a consortium of over 600 companies who collaboratively develop open specifications for high performance telecommunications and industrial computing applications, including the CompactPCI (cPCI) standard. CompactPCI products can be used in PXI Chassis and PXI products can be used in cPCI chassis with some limitations in each case.

Mission

Founded in 1994, PICMG's original mission was to extend the PCI standard, as approved by the PCI Special Interest Group (PCISIG) for computer systems such as PCI/ISA, PCI/EISA and the PCI/3U or 6U Eurocard form factor known as CompactPCI. PICMG continues to develop important extensions and improvements to CompactPCI.

Purpose

PICMG's purpose is to offer equipment vendors common specifications, thereby increasing availability and reducing costs and time to market. The PCI specifications provide a clear upgrade path for OEMs wishing to migrate to new designs.

As with PXISA, the PICMG is a not-for-profit organization.

PCISIG

The purpose of the PCI Special Interest Group (PCI-SIG) is to deliver a stable, straightforward and compatible standard for PCI devices.

The organization is important to PXI since the control interface is built on the PCI standard and uses common electrical devices to PCI cards, including the PCI Bridges incorporated into the PXI backplane.

USB

The USB Implementors Forum (USB-IF) promotes the Universal Serial Bus and maintains the specifications and a compliance program. USB 1.1, 2.0 (High Speed), and 3.0 (Super Speed) devices are available. USBTMC is a software interface standard created to support test and measurement applications and is now controlled by the IVI foundation.

VXI*plug&play* Systems Alliance

The VXI standard is controlled by the VXIbus Consortium. In 2002 the VXI*plug&play* Systems Alliance voted to become part of the IVI Foundation and was formally integrated into it in 2003.

USEFUL WEBSITES

AXIe Specifications	www.axiestandard.org
IEC Specifications	www.iec.org
IEEE Specifications	www.ieee.org
LXI Consortium	www.lxistandard.org
PCI Specifications	www.pcisig.com
Pickering Interfaces	www.pickeringtest.com
PICMG Specifications	www.picmg.org
PXI Specifications	www.pxisa.org
USB Specifications	www.usb.org
VISA Specifications	www.ivifoundation.org
VME Specifications	www.vita.com
IVI Foundation	www.ivifoundation.org
VXIbus Consortium	www.vxi.org

TERMINOLOGY

3U, 6U	Refers to the height of the module, the 6U modules being approximately twice the height of the 3U module. 1U is 44.45mm (1.75 inches)
API	Application Programming Interface
AXIe	AdvancedTCA Extensions for Instrumentation and Test
CompactPCI	A ruggedized version of PCI card conforming to PICMG 2.0 specification, providing superior mechanical performance and easier insertion or removal of cards.
GPIB	General Purpose Interface Bus, a standard used for interconnecting bench instruments using an 8 bit wide data system. Standard is defined by IEEE 488.
IVI	Interchangeable Virtual Instrument
J1, J2, J3	Connectors on PXI modules that mate to P1, P2 and P3. On 3U modules connectors J1 and J2 may mate to P1 and P2.
Local Bus	Bus that can be used to connect adjacent PXI modules in a chassis without the use of PXI features. The Bus can be used for analog or digital signals and is module defined.
PCI	Peripheral Component Interconnect, bus system commonly used in computers to provide additional functionality.
PCISIG	PCI Special Interest Group
PICMG	PCI Industrial Computer Manufacturers Group
PXI	PCI eXtensions for Instrumentation
Star Trigger	Fast trigger system driven from Slot 2 that provides low latency synchronized trigger to the Peripheral Modules.
Trigger Bus	A bus defined in PXI standard that can be used to trigger events. The triggers can be conditioned by software.
USB	Universal Serial Bus
VISA	Virtual Instrument Software Architecture
VXI	VME Extensions for Instrumentation
P1	Backplane connector on a chassis carrying the 32-bit PCI bus.
P2	Backplane connector on the chassis carrying the 64-bit PCI bus and PXI specific features.
P3	Reserved connector whose use is undefined by the PXI standard and can be fitted to 6U Chassis.
P4	Connector used on 6U Chassis that allows 3U cards to be fitted in 6U Slots. It provides the same functions as P1.
P5	Connector used on 6U Chassis that allows 3U cards to be fitted in 6U slots. It provides the same functions as P2.
System Slot or Slot 1	Slots on the left hand side of a PXI Chassis reserved for use by the system controller of a PXI Chassis (for PXIe the controller can be embedded elsewhere).

详细资料？请通过sales@hkaco.com联系我们。

广州：400 999 3848
上海：021-31215998
北京：010-57815068

HongKe 虹科
www.hkaco.com

广州虹科电子科技有限公司
测试测量和自动化产品 | 定制 | 培训
华南理工大学国家大学科技园2-504

www.hkaco.com