

---

Keithley PIO Series

# Using DriverLINX with Your Hardware

**KEITHLEY**

Information in this document is subject to change without notice. The software described in this document is furnished under a license agreement. The software may be used or copied only in accordance with the terms of the agreement.

SCIENTIFIC SOFTWARE TOOLS, INC. SHALL NOT BE LIABLE FOR ANY SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES RELATED TO THE USE OF THIS PRODUCT. THIS PRODUCT IS NOT DESIGNED WITH COMPONENTS OF A LEVEL OF RELIABILITY SUITABLE FOR USE IN LIFE SUPPORT OR CRITICAL APPLICATIONS.

This document may not, in whole or in part, be copied, photocopied, reproduced, translated or reduced to any electronic medium or machine readable form without prior written consent from Scientific Software Tools, Inc.

Keithley PIO Series: Using DriverLINX with Your Hardware  
Copyright © 1998-2000 by Scientific Software Tools, Inc.  
All rights reserved.

Third Printing.  
SST 21-1200-1

DriverLINX, SSTNET, and LabOBJX are registered trademarks and DriverLINX/VB is a trademark of Scientific Software Tools, Inc.  
MetraByte is a trademark of Keithley Instruments, Inc.  
Microsoft and Windows are registered trademarks and Visual C++ and Visual Basic are trademarks of Microsoft Corporation.  
Borland is a registered trademark and Borland C++ and Delphi are trademarks of Borland International, Inc.  
All other brand and product names are trademarks or registered trademarks of their respective companies.

# Contents

<b>Preface</b>	<b>5</b>
Software License and Software Disclaimer of Warranty.....	5
About DriverLINX.....	7
About This User's Guide .....	7
Conventions Used in This Manual .....	9
<b>Configuring the PIO Series</b>	<b>11</b>
Introduction.....	11
Configure DriverLINX Device Dialog.....	11
Device Subsystem Page .....	13
Digital Input Subsystem Page .....	16
Digital Output Subsystem Page .....	18
<b>Programming the PIO Series</b>	<b>21</b>
Selecting an API.....	21
Choosing DriverLINX .....	21
Choosing Hardware Direct I/O .....	22
Choosing ActiveX Direct I/O .....	23
Installing an API .....	23
Installing DriverLINX .....	24
Installing the ActiveX Direct I/O Interface.....	24
Programming an API.....	24
Programming DriverLINX.....	24
<b>Using the PIO Series with DriverLINX</b>	<b>29</b>
References.....	29
<b>Uninstalling DriverLINX</b>	<b>31</b>
How do I uninstall DriverLINX? .....	31
<b>Troubleshooting</b>	<b>33</b>
Solving Problems .....	33
Solving Problems Installing Drivers .....	33
Solving Problems Configuring the Drivers .....	33
Solving Problems Loading Drivers .....	34
Generating a DriverLINX Configuration Report .....	37
What is in the Report? .....	37
How do I Generate the Report? .....	37
<b>Glossary of Terms</b>	<b>38</b>



# Preface

---

## Software License and Software Disclaimer of Warranty

This is a legal document which is an agreement between you, the Licensee, and Scientific Software Tools, Inc. By opening this sealed diskette package, Licensee agrees to become bound by the terms of this Agreement, which include the Software License and Software Disclaimer of Warranty.

This Agreement constitutes the complete Agreement between Licensee and Scientific Software Tools, Inc. If Licensee does not agree to the terms of this Agreement, do not open the diskette package. Promptly return the unopened diskette package and the other items (including written materials, binders or other containers, and hardware, if any) that are part of this product to Scientific Software Tools, Inc. for a full refund. No refunds will be given for products that have opened disk packages or missing components.

### Licensing Agreement

**Copyright.** The software and documentation is owned by Scientific Software Tools, Inc. and is protected by both United States copyright laws and international treaty provisions. Scientific Software Tools, Inc. authorizes the original purchaser only (Licensee) to either (a) make one copy of the software solely for backup or archival purposes, or (b) transfer the software to a single hard disk only. The written materials accompanying the software may not be duplicated or copied for any reason.

**Trade Secret.** Licensee understands and agrees that the software is the proprietary and confidential property of Scientific Software Tools, Inc. and a valuable trade secret. Licensee agrees to use the software only for the intended use under this License, and shall not disclose the software or its contents to any third party.

**Copy Restrictions.** The Licensee may not modify or translate the program or related documentation **without the prior written consent of Scientific Software Tools, Inc.** All modifications, adaptations, and merged portions of the software constitute the software licensed to the Licensee, and the terms and conditions of this agreement apply to same. Licensee may not distribute copies, including electronic transfer of copies, of the modified, adapted or merged software or accompanying written materials to others. Licensee agrees not to reverse engineer, decompile or disassemble any part of the software.

Unauthorized copying of the software, including software that has been modified, merged, or included with other software, or of the written materials is expressly forbidden. Licensee may not rent, transfer or lease the software to any third parties. Licensee agrees to take all reasonable steps to protect Scientific Software Tools' software from theft, disclosure or use contrary to the terms of the License.

**License.** Scientific Software Tools, Inc. grants the Licensee only a non-exclusive right to use the serialized copy of the software on a single terminal connected to a single computer. The Licensee may not network the software or use it on more than one computer or computer terminal at the same time.

**Term.** This License is effective until terminated. This License will terminate automatically without notice from Scientific Software Tools, Inc. if Licensee fails to comply with any term or condition of this License. The Licensee agrees upon such termination to return or destroy the written materials and all copies of the software. The Licensee may terminate the agreement by returning or destroying the program and documentation and all copies thereof.

## Limited Warranty

Scientific Software Tools, Inc. warrants that the software will perform substantially in accordance with the written materials and that the program disk, instructional manuals and reference materials are free from defects in materials and workmanship under normal use for 90 days from the date of receipt. All express or implied warranties of the software and related materials are limited to 90 days.

Except as specifically set forth herein, the software and accompanying written materials (including instructions for use) are provided "as is" without warranty of any kind. Further, Scientific Software Tools, Inc. does not warrant, guarantee, or make any representations regarding the use, or the results of the use, of the software or written materials in terms of correctness, accuracy, reliability, currentness, or otherwise. The entire risk as to the results and performance of the software is assumed by Licensee and not by Scientific Software Tools, Inc. or its distributors, agents or employees.

**EXCEPT AS SET FORTH HEREIN, THERE ARE NO OTHER WARRANTIES, EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, WITH RESPECT TO THE SOFTWARE, THE ACCOMPANYING WRITTEN MATERIALS, AND ANY ACCOMPANYING HARDWARE.**

**Remedy.** Scientific Software Tools' entire liability and the Licensee's exclusive remedy shall be, at Scientific Software Tools' option, either (a) return of the price paid or (b) repair or replacement of the software or accompanying materials. In the event of a defect in material or workmanship, the item may be returned within the warranty period to Scientific Software Tools for a replacement without charge, provided the licensee previously sent in the limited warranty registration board to Scientific Software Tools, Inc., or can furnish proof of the purchase of the program. This remedy is void if failure has resulted from accident, abuse, or misapplication. Any replacement will be warranted for the remainder of the original warranty period.

**NEITHER SCIENTIFIC SOFTWARE TOOLS, INC. NOR ANYONE ELSE WHO HAS BEEN INVOLVED IN THE CREATION, PRODUCTION, SALE OR DELIVERY OF THIS PRODUCT SHALL BE LIABLE FOR ANY DIRECT, INDIRECT, CONSEQUENTIAL, OR INCIDENTAL DAMAGES (INCLUDING DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION, LOSS OF BUSINESS INFORMATION AND THE LIKE) ARISING OUT OF THE USE OF OR THE INABILITY TO USE SUCH PRODUCT EVEN IF SCIENTIFIC SOFTWARE TOOLS HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. BECAUSE SOME JURISDICTIONS DO NOT ALLOW THE EXCLUSION OR LIMITATION OF LIABILITY FOR CONSEQUENTIAL OR INCIDENTAL DAMAGES, OR LIMITATIONS ON DURATION OF AN IMPLIED WARRANTY, THE ABOVE LIMITATIONS MAY NOT APPLY TO LICENSEE.**

This agreement is governed by the laws of the Commonwealth of Pennsylvania.

---

# About DriverLINX

Welcome to DriverLINX® for Microsoft® Windows™, the high-performance real-time data-acquisition device drivers for Windows application development.

DriverLINX is a language- and hardware-independent application-programming interface designed to support hardware manufacturers' high-speed analog, digital, and counter/timer data-acquisition boards in Windows. DriverLINX is a multi-user and multitasking data-acquisition resource manager providing more than 100 services for foreground and background data acquisition tasks.

Included with your DriverLINX package are the following items:

- The DriverLINX API DLLs and drivers supporting your data-acquisition hardware
- PIO Panel, a DriverLINX program that verifies the installation and configuration of DriverLINX for your digital input/output board
- Learn DriverLINX, an interactive learning and demonstration program for DriverLINX that includes a Digital Storage Oscilloscope
- Source code for the sample programs
- The DriverLINX Application Programming Interface files for your compiler
- DriverLINX On-line Help System
- *DriverLINX 4.0 Installation and Configuration Guide*
- *DriverLINX Digital I/O Programming Guide*
- *DriverLINX Technical Reference Manual*
- Supplemental Documentation on DriverLINX and your data acquisition hardware

---

## About This User's Guide

The purpose of this manual is to help you quickly learn how to configure and use the hardware features of Keithley's PIO Series boards with DriverLINX.

- For help installing and configuring your hardware and DriverLINX, please see the manual that accompanied your hardware and the *DriverLINX 4.0 Installation and Configuration Guide* for your version of Windows.
- For more information on the DriverLINX API, please see the *DriverLINX Technical Reference Manual*.
- For additional help programming your board, please examine the source code examples on the Distribution Disks.

This manual contains the following chapters:

### **Configuring the PIO Series**

Shows how to configure the PIO Series using the *Configure DriverLINX Device* dialog box.

## **Using the PIO Series with DriverLINX**

Shows how to set up DriverLINX with the *Edit Service Request* dialog box to use PIO Series hardware features.



---

## Conventions Used in This Manual

The following notational conventions are used in this manual:

- A round bullet (●) identifies itemized lists.
- Numbered lists indicate a step-by-step procedure.
- DriverLINX Application Programming Interface and Windows macro and function names are set in bold when mentioned in the text.
- **DriverLINX** indicates the exported function name of the device driver DLL while DriverLINX indicates the product as a whole.
- DriverLINX Application Programming Interface identifiers, menu items, and Dialog Box names are italicized when mentioned in the text.
- *Italics* are used for emphasis.
- Source code and data structure examples are displayed in Courier typeface and bounded by a box with a single line.

Code

- A box with a double line bounds tables of information.

Tables

*Concept*

- Important concepts and notes are printed in the left margin.



# Configuring the PIO Series

---

## Introduction

The installation program provides general instructions for installing and configuring DriverLINX. This manual explains the steps and special features that apply to Keithley's PIO Series boards.


Installing and configuring DriverLINX for the Keithley PIO Series boards requires three steps:

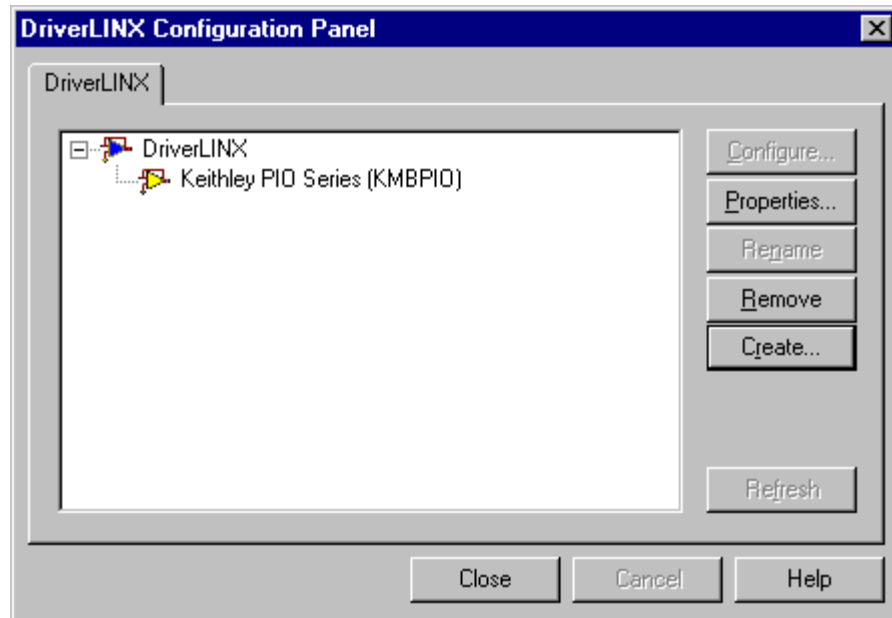
1. **Install DriverLINX.** Follow the instructions given by the installation program. The *Read Me First* instructions explain the components and drivers you can install.
2. **Configure DriverLINX.** See “Configure DriverLINX Device Dialog” on page 11 for configuration options specific to a Keithley PIO Series model.
3. **Install your PIO hardware,** read and follow the instructions in your hardware manual.

---

## Configure DriverLINX Device Dialog

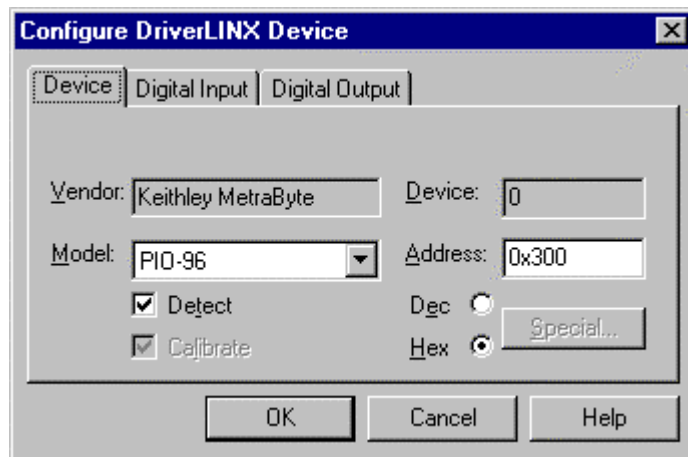
DriverLINX uses a standardized configuration protocol for all data-acquisition hardware. Configuration assigns a port address, interrupt resources and a DriverLINX Logical Device number to a specific PIO Series board in your computer.

The installation program automatically starts the *DriverLINX Configuration Panel*. To start it again later, use the shortcut on the Windows Start Menu or click here .



When you click the *Create...* or *Configure...* buttons on the *DriverLINX Configuration Panel*, DriverLINX displays the *Configure DriverLINX Device* dialog. The dialog has a page for each subsystem on a Keithley PIO Series model. The following sections describe your choices in configuring DriverLINX to work with your board.

## Device Subsystem Page



Use the Device subsystem page to tell DriverLINX the model name, address and, optionally, to disable detection of your PIO Series board.

### ***Vendor***

The Vendor property displays “Keithley MetraByte, Inc.” It is a read-only property.

### ***Device***

The Device property designates the Logical Device you are configuring. It is a read-only property. To change it, first save (**OK**) or quit (**Cancel**) the current configuration. Then select or create a new Logical Device using the *DriverLINX Configuration Panel*.

### ***Model***

The Model property selects or indicates the model of the board you’re configuring.

### ***Windows NT***

Select one of the following models:

- PIO-12
- PIO-24
- PIO-32I/O
- PIO-32IN
- PIO-32OUT
- PIO-96
- PIO-SSR-24
- PIO-SSR-48
- PIO-SSR-120
- PIO-HV
- PDISO-8
- REL-16
- Intel 8255

The selection “Intel 8255” will work with most digital boards that use an 8-bit ISA interface.

### **Windows 95/98**

Under Windows 95/98, DriverLINX displays the model you chose during installation. To install a different model, cancel the configuration and run *Add New Hardware* from the Windows Control Panel.

### **Address**

### **Windows NT**

The *Address* property records the I/O port address for the board. The default address used by DriverLINX is 768 decimal or 0x300 hex. If you have another peripheral board at that address, select a different base address. Note: consult your hardware manual for the number of consecutive free addresses you need.

### **Windows 95/98**

Under Windows 95/98, *Add New Hardware* automatically selects an appropriate address. To change the address, see “Using the Windows 95/98 Device Manager” on page 14.

### **Dec**

This radio button converts the *Address* property to decimal.

### **Hex**

This radio button converts the *Address* property to hexadecimal.

### **Detect**

The *Detect* property enables and disables DriverLINX’s hardware detection and testing algorithms. When you create a Logical Device, DriverLINX enables detection and testing. After loading the device successfully for the first time, DriverLINX automatically disables subsequent detection and testing.

You may need to disable detection yourself if your hardware has malfunctioning I/O ports that you’re not using.

### **Calibrate**

The *Calibrate* property enables and disables hardware auto-calibration. This option is grayed-out for the PIO Series because it does not support automatic calibration.





### **Special...**

The *Special...* button allows configuration of hardware-specific options. Keithley’s PIO Series does not need any special settings.

### **Using the Windows 95/98 Device Manager**

Under Windows 95/98, DriverLINX uses the address and interrupt settings maintained by the Windows Device Manager.

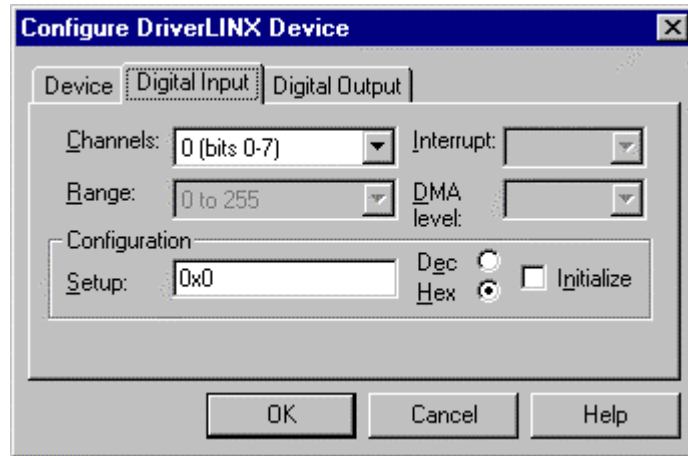
To view or change the settings for your board using the Device Manager:

1. Start the Device Manger by right-clicking on My Computer and selecting *Properties* or click here .
2. Click the *Device Manger* tab.
3. Click the  next to  DriverLINX drivers , if necessary to expand the list.
4. Under *DriverLINX drivers*, select the entry for your board. (It may or may not have  next to it.)



5. Click the *Properties* button.
6. On the board's property page, click the *Resources* tab.
7. To change a setting, select it under *Resource Type* and click the *Change Setting* button. Windows will guide you in selecting an appropriate value.
8. When you are done, click *OK* to close the board's property page.
9. The board's address switches must match the address setting you select. If necessary shut down your computer and reposition them as described in your hardware manual.
10. Restart Windows to load the Logical Device for your board using the new settings.

## Digital Input Subsystem Page



Use the Digital Input subsystem page to change the default digital input/output port configuration and, for applicable models, to set the board's interrupt level.

### **Channels**

The *Channels* property allows you to select a Logical Channel for input/output configuration or viewing the channel's range.

DriverLINX defines Logical Channels starting with zero. Each model has a different number of channels and, on some models, the Digital Input Subsystem share channels with the Digital Output Subsystem. See "Mapping Logical Channels to Digital Hardware Channels" in the *Digital I/O Programming Guide* for more details.

### **Range**

The *Range* property specifies the supported digital input range for the selected Logical Channel. This is a read-only property.

### **Interrupt**

#### **Windows NT**

The *Interrupt* property selects an interrupt level for your board. Only a few models support hardware interrupts.

#### **Windows 95/98**

Under Windows 95/98, *Add New Hardware* automatically selects an appropriate interrupt level. See "Using the Windows 95/98 Device Manager" on page 14 for information on changing the setting.

### **DMA level**

The PIO Series does not use system DMA channels. DriverLINX disables this property and displays it as blank.

### **Configuration Setup**

The *Configuration Setup* property specifies the hardware configuration of the digital I/O channels. See "Definition of Configuration Values" in the *Digital I/O Programming Guide* for more information.



### ***Dec***

This radio button converts the *Configuration Setup* property to decimal.

### ***Hex***

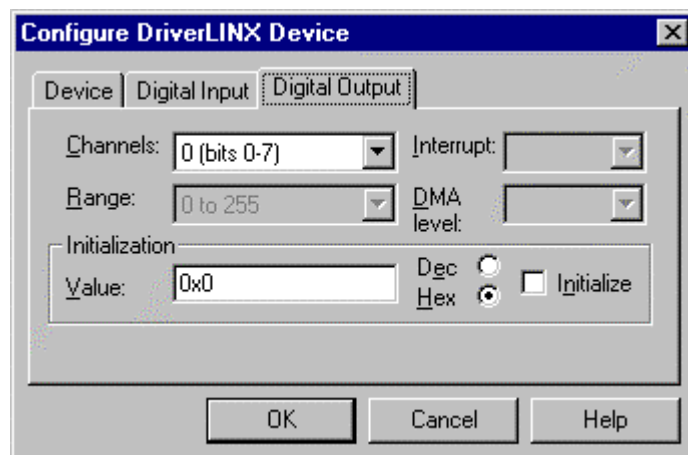
This radio button converts the *Configuration Setup* property to hexadecimal.

### ***Initialize***

Checking the *Initialize* check box instructs DriverLINX to use the *Configuration Setup* property to configure the selected digital I/O channel upon device initialization.

Applications can later change a channel's configuration. (See "Configuring a Digital Port at Run Time" in the *Digital I/O Programming Guide*.)

## Digital Output Subsystem Page



Use the Digital Output subsystem page to change the default digital output port initialization values.

### **Channels**

The *Channels* property allows you to select a Logical Channel for initialization or viewing the channel's range.

### **Range**

The *Range* property displays the supported digital output range for the selected Logical Channel. This is a read-only property.

### **Interrupt**

The PIO Series does not use interrupts for digital output. DriverLINX disables this property and displays it as blank.

### **DMA level**

The PIO Series does not use system DMA channels. DriverLINX disables this property and displays it as blank.

### **Initialization Value**

The *Initialization Value* property specifies the digital output value DriverLINX will write to the selected Logical Channel on hardware initialization. DriverLINX only writes this value if you enable the *Initialize* check box. By default, DriverLINX uses the hardware-defined initialization values if the *Initialize* check box is not checked. For the PIO Series, the default digital output value is zero.

### **Dec**

This radio button converts the *Initialization Value* property to decimal.

### **Hex**

This radio button converts the *Initialization Value* property to hexadecimal.

## ***Initialize***

Checking the *Initialize* check box instructs DriverLINX to use the *Initialization Value* property, rather than the default value, for digital output port initialization.



# Programming the PIO Series

---

## Selecting an API

The PIO Series supports two different device driver interfaces to best match your programming needs. The supported driver interfaces are

- **DriverLINX**—An interface that is hardware and operating system independent and supports multitasking, multithreading applications.
- **Direct I/O**—an interface that is operating system independent and supports single-tasking, single-threaded access to an Intel 8255-like function call interface.
- **ActiveX Direct I/O**—an interface that is operating system independent and supports single-tasking, single-threaded access to an Intel 8255-like function call interface.

Keithley recommends the using the DriverLINX interface for maximum portability and versatility, but the other interface is useful in special circumstances. Please review the following sections on the advantages and disadvantages of each interface.

Note: On Windows 95/98, you can also write directly to the board using CPU I/O instructions, but neither Microsoft nor Keithley recommend this technique. Windows NT does *not* support direct hardware access using CPU I/O instructions.

## Choosing DriverLINX

Keithley highly recommends selecting the DriverLINX interface for your applications. The DriverLINX interface has the following advantages:

- **Hardware independence**—DriverLINX supports ISA, PCMCIA, and PCI digital I/O boards with a common interface as well as digital ports on analog I/O and counter/timer boards.
- **Operating system independence**—DriverLINX supports Windows 95/98 and Windows NT with a common interface.
- **Shared hardware access**—DriverLINX allows multiple processes or threads to cooperatively share hardware resources.
- **Multitasking, multithreading support**—DriverLINX provides the synchronization and coordination for multitasking, multithreading applications to safely access shared hardware resources.

- **Portability**—DriverLINX supports older and newer hardware with a common interface for 16- and 32-bit applications on Windows 95/98 and Windows NT.
- **Versatility**—DriverLINX supports a wide variety of programming styles and languages including C/C++, Visual Basic, Delphi, and LabVIEW.
- **Background I/O**—DriverLINX supports background I/O using interrupts or other techniques so your application can overlap processing and data acquisition.
- **Full hardware support**—DriverLINX supports *all* the hardware's features.

The DriverLINX interface has a few disadvantages compared to the alternative interfaces:

- **Software modifications**—Existing applications will require rewriting to use DriverLINX.
- **No direct hardware access**—Before Windows NT, Microsoft discouraged, but nevertheless allowed, direct application program access to hardware.
- **Nibble configuration**—Some models support configurations of Port C with one input nibble and one output nibble. With the DriverLINX interface, this requires the use of the Configure DriverLINX Device dialog.
- **Speed of single-value I/O**—Applications, especially those ported from DOS, that depend on rapid software polling of I/O ports will notice that the operating system time cost for multitasking, multithreading synchronization is higher than direct hardware access.

While writing your application to use a multi-product, portable API is the best long-term solution, you should consider the alternative API *only* for special purpose or short-term needs.

## Choosing Hardware Direct I/O

The Direct Hardware I/O Interface simulates the hardware behavior of Intel 8255-based products, such as the PIO24 and PIO96, and the KMBPIO Series. This interface has the following advantages:

- **Direct I/O support**—This interface supports existing applications that write directly to the digital I/O's hardware registers using the x86 CPU's **inb** and **outb** instructions.
- **Intel 8255 emulation**—This interface simulates the I/O address map and a subset of the programming protocols of Intel's 8255 chip.
- **Fast hardware access**—This interface provides the fastest possible access to the hardware registers.
- **MS-DOS window support**—This interface will host 16- and 32-bit console applications running in an MS-DOS window.

The Hardware Direct I/O Interface has a few, significant disadvantages:

- **Win 9x only**—The I/O Emulation interface only runs on Windows 95/98. Windows NT does not support direct I/O without using a special port I/O driver available from Keithley's Web site at [www.keithley.com](http://www.keithley.com).
- **Non-exclusive hardware access**—This interface does not synchronize or coordinate hardware access among threads or processes.
- **Incompatible with DriverLINX**—Applications cannot use this interface with DriverLINX either in the same or another application.
- **Supports subset of hardware features**—This interface does not support using interrupts or PIO-series hardware features not available with Intel 8255-based products.

Keithley does not recommend using this interface for developing new applications.

## Choosing ActiveX Direct I/O

The ActiveX Direct I/O interface uses the methods of an ActiveX Automation object to access the KMBPIO hardware. This interface has the following advantages:

- **Operating system independence**—Both Windows 95/98 and Windows NT support this interface.
- **Intel 8255 emulation**—This interface simulates the I/O address map and programming protocols of Intel's 8255 chip.
- **Fast hardware access**—This interface provides faster access to the hardware registers than is available with the DriverLINX interface.
- **Versatility**—Most Windows compilers and scripting languages support ActiveX Automation objects.
- **Speed of single-value I/O**—The I/O performance of this interface is almost as fast as the Hardware Direct I/O interface when using early binding to the interface.
- **Board-relative addressing**—This interface automatically determines the actual hardware I/O address unlike the Hardware Direct I/O interface.

The ActiveX Direct I/O interface also has disadvantages:

- **Non-exclusive hardware access**—This interface does not synchronize or coordinate hardware access among threads or processes.
- **Incompatible with DriverLINX**—Applications cannot use this interface with DriverLINX either in the same or another application.
- **Supports subset of hardware features**—This interface does not support using interrupts.
- **ActiveX interface**—Using Automation objects in some C++ compilers is more difficult than in Visual Basic.

---

## Installing an API

To install a PIO programming interface, select one of the following topics for more information.

## Installing DriverLINX

The normal installation process automatically installs both the DriverLINX and Direct I/O APIs. Follow the instructions in the DriverLINX CD installer.

## Installing the ActiveX Direct I/O Interface

The normal installation process automatically installs both the DriverLINX and Direct I/O APIs. Follow the instructions in the DriverLINX CD installer.

### *Installing the ActiveX Direct I/O Driver*

The normal DriverLINX installation automatically installs the Direct I/O Driver. This driver is a registered ActiveX object that resides in KDIGIO.DLL.

This driver requires that you install the DriverLINX kernel drivers as it uses their services. On Win NT, the kernel driver is KMBPIO.SYS. On Win 95/98, it is KMBPIO.VXD. Follow the normal DriverLINX installation and configuration procedures to install these kernel drivers.

### *Configuring the ActiveX Direct I/O Driver*

The Direct I/O Driver requires that you configure a DriverLINX Logical Device using the DriverLINX Configuration Panel utility. The Direct I/O driver does not require any separate configuration.

---

## Programming an API

To use a PIO programming interface, select one of the following topics for more information.

### Programming DriverLINX

The DriverLINX API is extensively documented in this manual and in the *Digital I/O Programming Guide* and the *DriverLINX or DriverLINX/VB Technical Reference Manuals*.

The ActiveX Direct I/O Interface is an ActiveX Automation object hosting one interface. This control is compatible with Microsoft Visual C++ and Visual Basic as well as other ActiveX-hosting languages.

The methods of the control's interface, IKDigitalIo, emulate the 8-bit port I/O behavior of an Intel 8255 chip. Consult an Intel hardware manual or the programming section of your PIO-Series hardware's *User's Manual*.

### *IKDigitalIo Intel 8255 Interface*

The following syntax descriptions are shown in C/C++. For Visual Basic, use the Object Browser to see the VB syntax. Also see "Using the Direct I/O Driver in Visual Basic" on page 25.

- HRESULT OpenDevice(BSTR Driver, long Device);  
Opens a digital I/O device.  
Driver is "KMBPIO" to open a Keithley PIO Series board.



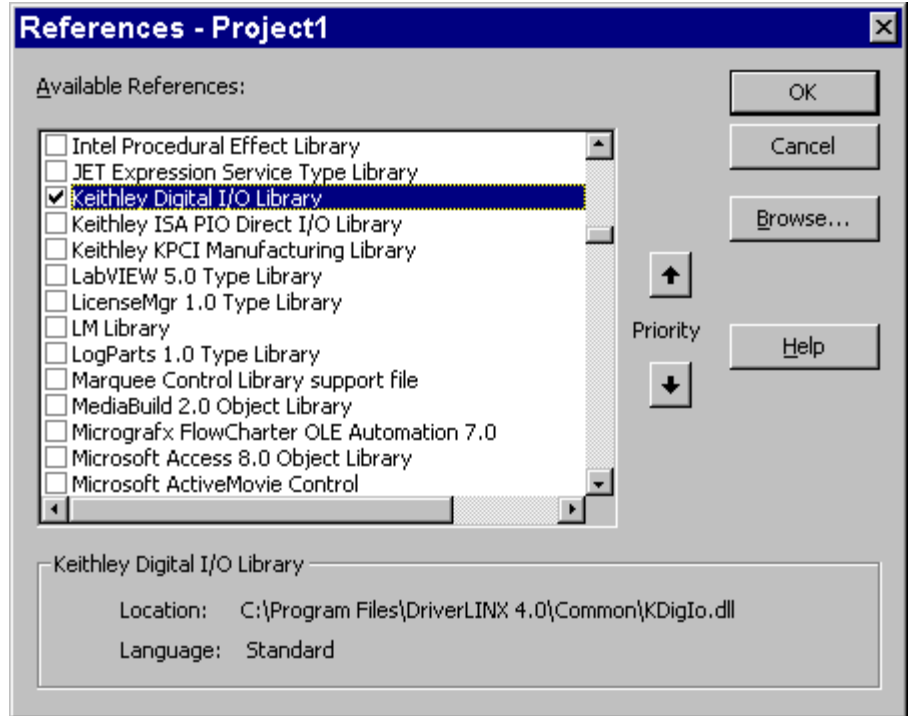
Device is the DriverLINX Logical Device number of the KMBPIO board to open.

- **HRESULT CloseDevice();**  
Closes a previously opened KMBPIO device.  
The number of CloseDevice calls must match the number of OpenDevice calls.
- **HRESULT Read(short Offset, unsigned char \* Result);**  
Reads an 8-bit value from the KMBPIO board.  
Offset is the value of the register to read relative the base I/O address.  
Result is the value read from the register.
- **HRESULT Write(short Offset, unsigned char Value);**  
Writes an 8-bit value to the KMBPIO board.  
Offset is the value of the register to write relative the base I/O address.  
Value is the output to write to the register.
- **HRESULT OpenCount(long \*Count);**  
Returns the number of times a client has called OpenDevice for this Logical Device.  
Count is the address of the value for the result.
- **HRESULT Device(long \*LogicalDevice);**  
Returns the Logical Device number if the device is open. Otherwise it returns -1.  
LogicalDevice is the address of the value for the result.
- **HRESULT Driver(BSTR \*Name);**  
Returns the DriverLINX class driver name if any device is open.  
Otherwise it returns an empty string.  
Name is the address of a string value for the result.

### ***Using the Direct I/O Driver in Visual Basic***

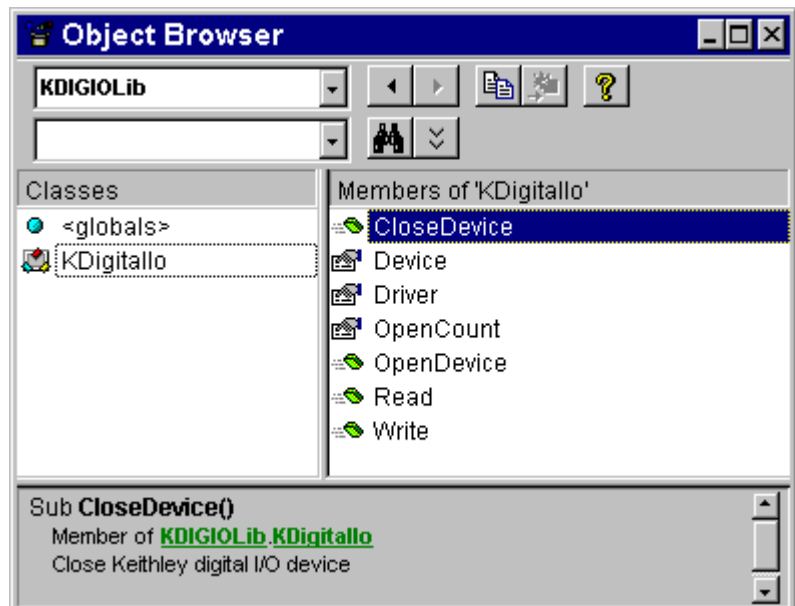
The following instructions are for Version 5 of Microsoft Visual Basic. If you have a different version, please consult the Visual Basic documentation.

1. From VB's "Project" menu, select "References..."
2. Scroll through the "Available References", looking for "Keithley Digital I/O Library".



Check “Keithley Digital I/O Library” and then click OK.

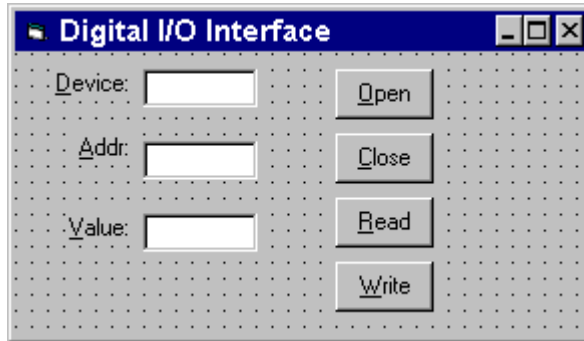
3. From VB’s “View” menu, select “Object Browser” to display a list of libraries.
4. Search for “KDIGIOLib”.



5. The Object Browser shows the VB syntax for the KDigitalIo (see “IKDigitalIo Intel 8255 Interface” on page 24) class.

## Creating a Simple Visual Basic Application

1. Create a Visual Basic form:



2. Add the following source code to declare, open, close, read, and write a PIO device.

```
Option Explicit
Dim KPIO As New KDigitalIo

Private Sub btClose_Click()
    KPIO.CloseDevice
End Sub

Private Sub btOpen_Click()
    KPIO.OpenDevice ("KMBPIO", Val(tbDevice))
End Sub

Private Sub btRead_Click()
    tbValue = Hex(KPIO.Read(Val(tbAddr)))
End Sub

Private Sub btWrite_Click()
    Dim v As Byte

    v = Val(tbValue)
    KPIO.Write Val(tbAddr), v
End Sub
```



# Using the PIO Series with DriverLINX

---

## References

Please see the following references for information on programming your Keithley PIO Series board with DriverLINX:

- ***Digital I/O Programming Guide***  
Shows you how to set up and use PIO Series hardware features with DriverLINX.
- ***DriverLINX Technical Reference Manual***  
Describes the DriverLINX Application Program Interface (API) for the DLL interface using C++.
- ***DriverLINX/VB Technical Reference Manual***  
Describes the API for the ActiveX (OCX) interface using VB or Delphi.
- **DriverLINX examples for your programming language**  
They are in the subfolders of your DriverLINX installation folder or on the original distribution media.



# Uninstalling DriverLINX

---

## How do I uninstall DriverLINX?

DriverLINX consists of three separate component installations:

- DriverLINX for the Keithley PIO Series
- DriverLINX Programming Interfaces
- DriverLINX Documentation

You can uninstall the last two installations at any time without interfering with compiled applications that require DriverLINX drivers. To uninstall the latter components, run the “Add/Remove Programs” tool in the Windows Control Panel.

To uninstall DriverLINX drivers for the Keithley PIO Series, you must

- Disable the DriverLINX driver.
- Shut down your computer to remove the hardware.
- Reboot your computer to unload the driver.
- Run the DriverLINX uninstall program.

### ***How to Disable a DriverLINX Driver in Windows NT***

1. From the Windows Start menu, select “Settings”, then “Control Panel”. Left click on the DriverLINX Configuration icon in the Control Panel.
2. Select the PIO devices you want to disable.
3. Right click on each device and select “Disabled” on the popup menu.
4. Repeat steps 2-3 for each PIO card that you are uninstalling.
5. Close the DriverLINX Configuration Panel.
6. When finished, shut down your computer and physically remove any installed PIO hardware.
7. Reboot Windows.
8. To finish uninstalling, see “How to Remove DriverLINX for the Keithley PIO Series” on page 32.

## ***How to Disable a DriverLINX Driver in Windows 95/98***

1. From the Windows Start menu, select “Settings”, then “Control Panel”. Left click on the System icon in the Control Panel. Select the “Device Manager” tab in the System Properties dialog.
2. Left click the “+” icon next to “DriverLINX drivers” to display the installed Keithley PIO devices.



3. Select the PIO device you want to disable.
4. Click the “Remove” button.
5. In the “Confirm Device Removal” dialog, select “OK”.
6. Repeat steps 3-5 for each PIO card or driver that you are uninstalling.
7. When finished, click “Close”, shut down your computer, and physically remove any installed PIO hardware.
8. Reboot Windows.
9. To finish uninstalling, see “How to Remove DriverLINX for the Keithley PIO Series” on page 32.

## ***How to Remove DriverLINX for the Keithley PIO Series***

1. From the Windows Start menu, select “Settings”, then “Control Panel”. Left click on the Add/Remove Programs icon in the Control Panel.
2. Select “DriverLINX for Keithley PIO” in the Add/Remove Programs Properties dialog.
3. Click the “Add/Remove...” button.
4. Answer “Yes” to “Are you sure you want to remove ‘DriverLINX for Keithley PIO Series’ and all of its components?” in the Confirm File Deletion dialog.
5. The DriverLINX uninstall program will proceed.

---

The uninstall program will not remove the folder, “\DrvLINX4\System”. This folder contains copies of any \Windows\System or \Windows\System32 files that the original DriverLINX installation updated.

---



# Troubleshooting

---

## Solving Problems

Correct operation of your PIO hardware requires successful completion of four steps.

1. Windows finds free resources for the PIO board.
2. The PIO address switches are set to the assigned address resource.
3. You configure the PIO drivers using the DriverLINX Configuration Panel.
4. Windows loads the PIO drivers into memory.

If you are having a problem installing or configuring your PIO product, review the following notes. If these notes do not solve your problem, or your problem is not described, then contact technical support and fully describe your problem.

### Solving Problems Installing Drivers

The DriverLINX installation program runs a wizard that assists you in installing, registering and configuring the DriverLINX driver for your board. If you would like to repeat any steps with the wizard, click here [🔗](#).

### Solving Problems Configuring the Drivers

Windows 95/98 assigns hardware resources for the PIO, but you must still configure the PIO drivers before using them. The DriverLINX configuration requires that you select the hardware model of your PIO board.


On Windows NT, you must, also, manually enter the address and interrupt resource assignments. See “Configuring the PIO Series” on page 11 for more information.

## Solving Problems Loading Drivers

Before the PIO drivers can load, you must

1. Install the DriverLINX software.
2. Install the PIO hardware into your computer.
3. Configure DriverLINX.
4. Reboot your computer.

If you have not completed the above steps, please do so before proceeding.

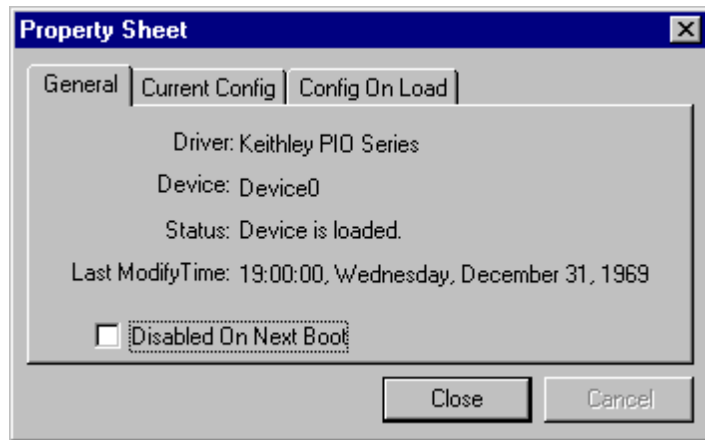
On Windows NT you must determine free hardware resources for the PIO using Windows NT Diagnostics . On Windows 95/98, the operating system will automatically assign hardware resources to the PIO cards. Automatic resource assignment can fail sometimes on

- Older PCI computers.
- Computers with ISA cards installed.
- Computers with no free hardware resources.

Sorting through all possibilities can be a challenge due to the sheer number of combinations of hardware designs, PC plug-in boards, and versions of Windows. The following sections will help you gather information about why a driver may have failed to load. This information is essential for you or technical support to solve your problem.

### ***Did the DriverLINX Driver Load?***

1. Run “DriverLINX Configuration” from Windows Control Panel.
2. Select the “DriverLINX” tab.
3. Click the “+” icon next to DriverLINX to expand the list of drivers, if necessary.
4. Select “Keithley PIO”. Click “+”, if necessary, to expand the list.
5. Select the line with the number of the Logical Device you configured. If the number does not exist, you did not configure the driver. See “Configuring the PIO Series” on page 11.
6. Click the “Properties...” button and then select the “General” tab.
7. Do you see “Status: Device is loaded”? If not, did you reboot the computer after configuring? If not, reboot now and repeat the above steps.



8. If you rebooted the computer after configuring and Windows did not load your device, see “Checking for Device Errors” on page 35.

### ***Checking for Device Errors***

When a DriverLINX kernel driver cannot load, it writes an explanation into the system event log. You can view this log under Windows 95/98 or Windows NT using the DriverLINX Event Viewer.

Windows 95/98 maintains additional driver information in the Device Manager. Also see “Getting More Driver Information on Windows 95/98” on page 35.

1. Run “DriverLINX Event Viewer” from the DriverLINX folder.
2. Click on the “+” icon next to “DriverLINX” in the left panel.
3. Select the abbreviation for your driver.
4. Does the first line in the right panel show a current error?
5. Double click on the error line to see more detail and an explanatory message.
6. If you cannot resolve the problem yourself, please provide this error information when contacting technical support.

### ***Getting More Driver Information on Windows 95/98***

Windows 95/98 reports additional information about device status using the Device Manager. To access this utility,

1. Right click on “My Computer” and then select “Properties”.
2. Select “Device Manager” and “View devices by type”.
3. Does “DriverLINX drivers” appear in the list? If not, see “Solving Problems Installing Drivers” on page 33.
4. Click the “+” next to “DriverLINX drivers”.
5. Does your PIO product appear in the list? If not, see “Solving Problems Installing Drivers” on page 33.
6. Does the icon next to your PIO product display an exclamation point (!)? If no, Windows has loaded your PIO driver.
7. Select the line with the “!” and then click “Properties”.

8. The General tab will show the reason why the driver did not load.
9. The Resources tab will show if Windows detected an unresolvable hardware conflict.

### ***Getting More Driver Information on Windows NT***

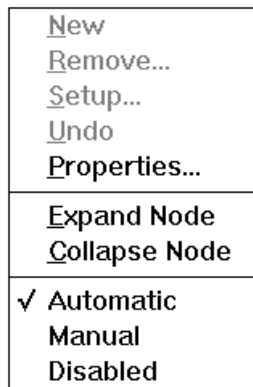
On Windows NT, the only reasons that a driver does not load are

- You did not install the driver software.
- You did not correctly configure the driver.
- You changed the driver startup parameters.

An incorrectly configured driver will report the reasons that it failed to load into the Windows Event Log. See “Checking for Device Errors” on page 35 for more information.

On Windows NT, DriverLINX drivers load automatically during system boot. An administrator can change the startup command for any NT driver to either “manual” or “disabled”.

1. Run “DriverLINX Configuration” from Windows Control Panel.
2. Select the “DriverLINX” tab.
3. Click the “+” icon next to DriverLINX to expand the list of drivers, if necessary.
4. Select “Keithley PIO”. Click “+”, if necessary, to expand the list.
5. Select the line with the number of the Logical Device that did not load.
6. Right click the mouse to see a popup menu.



7. Select “Automatic” to instruct Windows to load the driver the next time you reboot.

---


# Generating a DriverLINX Configuration Report

Your DriverLINX installation includes a troubleshooting tool that generates a report of your DriverLINX configuration. If you call Technical Support, after reading “Solving Problems” on page 33, they may ask you to generate and e-mail this report to help you solve installation and configuration problems.

## What is in the Report?

The troubleshooting tool analyzes your computer to obtain information about DriverLINX and operating system software that would assist Technical Support in troubleshooting a problem you are having. It includes information on DriverLINX files, environment variables, registry entries, hardware and the operating system.

## How do I Generate the Report?

You can easily generate the report by clicking this shortcut . Once the troubleshooting tool generates the report, you will have the opportunity to review it and make deletions, if desired, before e-mailing it to Technical Support. If you do not have direct access to e-mail, you can save the report to a disk file and send a copy later. A Technical Support engineer will guide you through these steps when you are asked to send a report.

# Glossary of Terms

## **A/D**

Abbreviation for Analog-to-Digital, a process that converts a continuous analog signal into a discrete digital approximation of the analog signal.

## **ADC**

Abbreviation for Analog-to-Digital Converter, the hardware that performs the A/D conversion.

## **API**

Abbreviation for Application Programming Interface. An API defines the syntax of the data structures and functions of software services.

## **DMA**

Abbreviation for Direct Memory Access, a technique where the system board can transfer data between a device and memory without using the CPU. In the PC, a standard chip on the system board controls the transfer.

## **IRQ**

Abbreviation for interrupt request. Peripheral hardware signals the CPU that it is ready to transfer data.

## **ISA**

Abbreviation for Industry Standard Architecture. A standard for the original IBM AT bus specification that defines the bus structure, CPU and support chip architecture, and the clock frequency of the ISA bus.

## **ISR**

Abbreviation for interrupt service routine, the software function inside a device driver that handles interrupt requests.

## **Logical Device**

DriverLINX's designation for a specific data-acquisition board inside your computer.

## **Service Request**

A DriverLINX object or data structure that completely defines a data-acquisition task.

## **Subsystem**

DriverLINX subdivides a general purpose data-acquisition device into six subsystems—Device, Analog Input, Analog Output, Digital Input, Digital Output, and Counter/Timer.

## **TTL**

Abbreviation for transistor-transistor logic, a family of digital logic elements.