

# KPXI Multi-Function Module

## Reference Manual

KPXI-DAQ-901-01 Rev. A / January 2007

ECA 42912

# WARRANTY

Keithley Instruments, Inc. warrants this product to be free from defects in material and workmanship for a period of 1 year from date of shipment.

Keithley Instruments, Inc. warrants the following items for 90 days from the date of shipment: probes, cables, rechargeable batteries, diskettes, and documentation.

During the warranty period, we will, at our option, either repair or replace any product that proves to be defective.

To exercise this warranty, write or call your local Keithley Instruments representative, or contact Keithley Instruments headquarters in Cleveland, Ohio. You will be given prompt assistance and return instructions. Send the product, transportation prepaid, to the indicated service facility. Repairs will be made and the product returned, transportation prepaid. Repaired or replaced products are warranted for the balance of the original warranty period, or at least 90 days.

## LIMITATION OF WARRANTY

This warranty does not apply to defects resulting from product modification without Keithley Instruments' express written consent, or misuse of any product or part. This warranty also does not apply to fuses, software, non-rechargeable batteries, damage from battery leakage, or problems arising from normal wear or failure to follow instructions.

THIS WARRANTY IS IN LIEU OF ALL OTHER WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR USE. THE REMEDIES PROVIDED HEREIN ARE BUYER'S SOLE AND EXCLUSIVE REMEDIES.

NEITHER KEITHLEY INSTRUMENTS, INC. NOR ANY OF ITS EMPLOYEES SHALL BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OF ITS INSTRUMENTS AND SOFTWARE EVEN IF KEITHLEY INSTRUMENTS, INC., HAS BEEN ADVISED IN ADVANCE OF THE POSSIBILITY OF SUCH DAMAGES. SUCH EXCLUDED DAMAGES SHALL INCLUDE, BUT ARE NOT LIMITED TO: COSTS OF REMOVAL AND INSTALLATION, LOSSES SUSTAINED AS THE RESULT OF INJURY TO ANY PERSON, OR DAMAGE TO PROPERTY.



A G R E A T E R M E A S U R E O F C O N F I D E N C E

**Keithley Instruments, Inc.**

Corporate Headquarters • 28775 Aurora Road • Cleveland, Ohio 44139  
440-248-0400 • Fax: 440-248-6168 • 1-888-KEITHLEY (534-8453) • [www.keithley.com](http://www.keithley.com)

# KPXI Multi-Function Module Reference Manual

©2007, Keithley Instruments, Inc.  
All rights reserved.  
Cleveland, Ohio, U.S.A.

Document Number: KPXI-DAQ-901-01 Rev. A / January 2007

## Manual Print History

The print history shown below lists the printing dates of all Revisions and Addenda created for this manual. The Revision Level letter increases alphabetically as the manual undergoes subsequent updates. Addenda, which are released between Revisions, contain important change information that the user should incorporate immediately into the manual. Addenda are numbered sequentially. When a new Revision is created, all Addenda associated with the previous Revision of the manual are incorporated into the new Revision of the manual. Each new Revision includes a revised copy of this print history page.

Revision A (Document Number KPXI-DAQ-901-01)..... January 2007

The following safety precautions should be observed before using this product and any associated instrumentation. Although some instruments and accessories would normally be used with non-hazardous voltages, there are situations where hazardous conditions may be present.

This product is intended for use by qualified personnel who recognize shock hazards and are familiar with the safety precautions required to avoid possible injury. Read and follow all installation, operation, and maintenance information carefully before using the product. Refer to the manual for complete product specifications.

If the product is used in a manner not specified, the protection provided by the product may be impaired.

The types of product users are:

**Responsible body** is the individual or group responsible for the use and maintenance of equipment, for ensuring that the equipment is operated within its specifications and operating limits, and for ensuring that operators are adequately trained.

**Operators** use the product for its intended function. They must be trained in electrical safety procedures and proper use of the instrument. They must be protected from electric shock and contact with hazardous live circuits.

**Maintenance personnel** perform routine procedures on the product to keep it operating properly, for example, setting the line voltage or replacing consumable materials. Maintenance procedures are described in the manual. The procedures explicitly state if the operator may perform them. Otherwise, they should be performed only by service personnel.

**Service personnel** are trained to work on live circuits, and perform safe installations and repairs of products. Only properly trained service personnel may perform installation and service procedures.

Keithley Instruments products are designed for use with electrical signals that are rated Measurement Category I and Measurement Category II, as described in the International Electrotechnical Commission (IEC) Standard IEC 60664. Most measurement, control, and data I/O signals are Measurement Category I and must not be directly connected to mains voltage or to voltage sources with high transient over-voltages. Measurement Category II connections require protection for high transient over-voltages often associated with local AC mains connections. Assume all measurement, control, and data I/O connections are for connection to Category I sources unless otherwise marked or described in the Manual.

Exercise extreme caution when a shock hazard is present. Lethal voltage may be present on cable connector jacks or test fixtures. The American National Standards Institute (ANSI) states that a shock hazard exists when voltage levels greater than 30V RMS, 42.4V peak, or 60VDC are present. A good safety practice is to expect that hazardous voltage is present in any unknown circuit before measuring.

Operators of this product must be protected from electric shock at all times. The responsible body must ensure that operators are prevented access and/or insulated from every connection point. In some cases, connections must be exposed to potential human contact. Product operators in these circumstances must be trained to protect themselves from the risk of electric shock. If the circuit is capable of operating at or above 1000 volts, no conductive part of the circuit may be exposed.

Do not connect switching cards directly to unlimited power circuits. They are intended to be used with impedance limited sources. NEVER connect switching cards directly to AC mains. When connecting sources to switching cards, install protective devices to limit fault current and voltage to the card.

Before operating an instrument, make sure the line cord is connected to a properly grounded power receptacle. Inspect the connecting cables, test leads, and jumpers for possible wear, cracks, or breaks before each use.

When installing equipment where access to the main power cord is restricted, such as rack mounting, a separate main input power disconnect device must be provided, in close proximity to the equipment and within easy reach of the operator.

For maximum safety, do not touch the product, test cables, or any other instruments while power is applied to the circuit under test. ALWAYS remove power from the entire test system and discharge any capacitors before: connecting or disconnecting cables or jumpers, installing or removing switching cards, or making internal changes, such as installing or removing jumpers.

Do not touch any object that could provide a current path to the common side of the circuit under test or power line (earth) ground. Always make measurements with dry hands while standing on a dry, insulated surface capable of withstanding the voltage being measured.


The instrument and accessories must be used in accordance with its specifications and operating instructions or the safety of the equipment may be impaired.


Do not exceed the maximum signal levels of the instruments and accessories, as defined in the specifications and operating information, and as shown on the instrument or test fixture panels, or switching card.


When fuses are used in a product, replace with same type and rating for continued protection against fire hazard.

Chassis connections must only be used as shield connections for measuring circuits, NOT as safety earth ground connections.

If you are using a test fixture, keep the lid closed while power is applied to the device under test. Safe operation requires the use of a lid interlock.

If a  screw is present, connect it to safety earth ground using the wire recommended in the user documentation.

The  symbol on an instrument indicates that the user should refer to the operating instructions located in the manual.

The  symbol on an instrument shows that it can source or measure 1000 volts or more, including the combined effect of normal and common mode voltages. Use standard safety precautions to avoid personal contact with these voltages.

The  symbol on an instrument shows that the surface may be hot. Avoid personal contact to prevent burns.

The  symbol indicates a connection terminal to the equipment frame.

The **WARNING** heading in a manual explains dangers that might result in personal injury or death. Always read the associated information very carefully before performing the indicated procedure.

The **CAUTION** heading in a manual explains hazards that could damage the instrument. Such damage may invalidate the warranty.

Instrumentation and accessories shall not be connected to humans.

Before performing any maintenance, disconnect the line cord and all test cables.

To maintain protection from electric shock and fire, replacement components in mains circuits, including the power transformer, test leads, and input jacks, must be purchased from Keithley Instruments. Standard fuses, with applicable national safety approvals, may be used if the rating and type are the same. Other components that are not safety related may be purchased from other suppliers as long as they are equivalent to the original component. (Note that selected parts should be purchased only through Keithley Instruments to maintain accuracy and functionality of the product.) If you are unsure about the applicability of a replacement component, call a Keithley Instruments office for information.

To clean an instrument, use a damp cloth or mild, water based cleaner. Clean the exterior of the instrument only. Do not apply cleaner directly to the instrument or allow liquids to enter or spill on the instrument. Products that consist of a circuit board with no case or chassis (e.g., data acquisition board for installation into a computer) should never require cleaning if handled according to instructions. If the board becomes contaminated and operation is affected, the board should be returned to the factory for proper cleaning/servicing.

# Table of Contents

---

<b>Section</b>	<b>Topic</b>	<b>Page</b>	
<b>1</b>	<b>Introduction</b> .....	1-1	
	Introduction.....	1-2	
	Features.....	1-2	
	Applications.....	1-2	
	Safety symbols and terms.....	1-2	
	Specifications.....	1-3	
	Unpacking and inspection.....	1-3	
	Inspection for damage.....	1-3	
	Shipment contents.....	1-3	
	Instruction manual.....	1-4	
	Repacking for shipment.....	1-4	
	Software introduction.....	1-4	
	Programming library KDAQ-DRVR.....	1-4	
	KDAQ-LVIEW LabVIEW® driver.....	1-4	
	<b>2</b>	<b>Installation</b> .....	2-1
		Introduction.....	2-2
		Handling precautions.....	2-2
Mechanical Drawing.....		2-2	
Configuration.....		2-2	
Plug and Play.....		2-2	
Configuration.....		2-3	
Troubleshooting.....		2-3	
Installation.....		2-3	
<b>3</b>		<b>Operation</b> .....	3-1
	Introduction.....	3-2	
	Signal Connections.....	3-2	
	Connectors Pin Assignment.....	3-2	
	Analog Input Signal Connection.....	3-7	
	Operation Theory.....	3-9	
	A/D Conversion.....	3-9	
	D/A Conversion.....	3-20	
	Digital I/O.....	3-26	
	General Purpose Timer/Counter Operation.....	3-27	
	Trigger Sources.....	3-30	
	User-controllable Timing Signals.....	3-34	
	Calibration.....	3-38	
	Loading Calibration Constants.....	3-38	
	Auto-calibration.....	3-38	
	Saving Calibration Constants.....	3-39	
	<b>Appendix</b>	<b>Topic</b>	<b>Page</b>
<b>A</b>	<b>KDAQ-DRVR User's Guide</b> .....	A-1	
	Introduction to KDAQ-DRVR.....	A-2	
	About the KDAQ-DRVR software.....	A-2	
	KDAQ-DRVR hardware support.....	A-2	
	KDAQ-DRVR language support.....	A-2	

Appendix	Topic	Page		
<b>A</b>	<b>KDAQ-DRVR User's Guide (continued)</b>			
	Fundamentals of building applications with KDAQ-DRVR .....	A-3		
	Microsoft® Visual Basic (Version 6.0) .....	A-3		
	Using Microsoft Visual Basic.NET .....	A-4		
	Microsoft Visual C/C++ .....	A-4		
	KDAQ-DRVR utilities for Win32 .....	A-5		
	KDAQ-DRVR configuration utility (configdrv) .....	A-5		
	KDAQ-DRVR data file converter utility (KiDAQCvt) .....	A-6		
	KDAQ-DRVR overview .....	A-6		
	General configuration function group .....	A-7		
	Analog input function group .....	A-7		
	Analog output function group .....	A-10		
	Digital input function group .....	A-12		
	Digital output function group .....	A-13		
	General timer/counter function group .....	A-13		
	DIO function group .....	A-13		
	SSI function group .....	A-14		
	Calibration function group .....	A-14		
	KDAQ-DRVR application hints .....	A-15		
	Analog input programming hints .....	A-16		
	Analog output programming hints .....	A-36		
	One-shot analog output programming scheme .....	A-36		
	Digital input programming hints .....	A-50		
	Digital output programming hints .....	A-51		
	DAQ event message programming hints .....	A-52		
	Continuous data transfer in KDAQ-DRVR .....	A-53		
	Continuous data transfer mechanism .....	A-53		
	Double-buffered AI/AO operation .....	A-53		
	Single-buffered versus double-buffered data transfer .....	A-54		
	Pre-trigger mode/middle-trigger data acquisition (AI) .....	A-54		
	<b>B</b>	<b>KDAQ-DRVR Function Reference</b> .....	B-1	
		Function description .....	B-2	
		Data types .....	B-2	
		Function reference .....	B-2	
		Status Codes .....	B-94	
		AI range codes .....	B-95	
		AI data format .....	B-97	
		DATA file format .....	B-97	
		Header .....	B-98	
		ChannelRange .....	B-99	
		Data Block .....	B-99	
		<b>C</b>	<b>KIDAQ®-LabVIEW Compatible Interface Guide</b> .....	C-1
			Introduction to KIDAQ®-LabVIEW .....	C-2
			Overview .....	C-2
			Using KIDAQ LabVIEW VIs in LabVIEW .....	C-2
KIDAQ LabVIEW Programming .....	C-3			
Device Driver Handling .....	C-4			
Windows XP/2000 Device Driver .....	C-4			
Driver Utility .....	C-4			
KIDAQ Utilities .....	C-4			
KIDAQ Registry/Configuration utility .....	C-4			
KIDAQ Device Browser .....	C-4			
KIDAQ LabVIEW VIs Overview .....	C-5			
Analog Input VIs .....	C-6			
Analog Output VIs .....	C-6			
Digital I/O VIs .....	C-7			
Timer/Counter VIs .....	C-7			
Calibration and Configuration VIs .....	C-8			
Error Handler VI .....	C-8			



Appendix	Topic	Page
<b>C</b>	<b>KIDAQ®-LabVIEW Compatible Interface Guide (continued)</b>	
	Distribution of Applications .....	C-8
	Windows XP/2000 .....	C-8
<b>D</b>	<b>KIDAQ®-LabVIEW Compatible Function Reference</b> .....	D-1
	Introduction .....	D-2
	Hardware support .....	D-2
	KPXI-DIO series: .....	D-2
	KPXI-DAQ series: .....	D-2
	Digitizer series: .....	D-2
	Analog input VIs .....	D-3
	Easy analog input VIs .....	D-3
	Intermediate analog input VIs .....	D-7
	Analog output VIs .....	D-21
	Easy analog output VIs .....	D-21
	Intermediate analog output VIs .....	D-24
	Advanced analog output VIs .....	D-32
	Digital I/O VIs .....	D-33
	Easy Digital I/O VIs .....	D-33
	Intermediate Digital I/O VIs .....	D-37
	Advanced Digital I/O VIs .....	D-45
	Counter VIs .....	D-46
	Easy Counter VIs .....	D-46
	Intermediate Counter VIs .....	D-50
	Advanced Counter VIs .....	D-63
	Calibration and Configuration VIs .....	D-67
	Calibration VIs .....	D-67
	Other Calibration and Configuration VIs .....	D-68
	Service VIs .....	D-70
	Error Codes .....	D-71
	AI Range Codes .....	D-73
	AI Data Format .....	D-76

**Service Form**

This page left blank intentionally.

# List of Figures

---

Section	Figure	Title	Page
2	Figure 2-1	PXI Layout of KPXI-DAQ .....	2-2
	Figure 2-2	Typical PXI module installation .....	2-4
	Figure 2-3	Device manager (successful installation) .....	2-5
3	Figure 3-1	CN1 pinout (KPXI-DAQ-64-3M/KPXI-DAQ-64-500K/ KPXI-DAQ-64-250K).....	3-2
	Figure 3-2	CN2 pinout (KPXI-DAQ-64-3M/KPXI-DAQ-64-500K/ KPXI-DAQ-64-250K).....	3-3
	Figure 3-3	CN1 pinout (KPXI-DAQ-96-3M).....	3-4
	Figure 3-4	CN2 pinout (KPXI-DAQ-96-3M).....	3-5
	Figure 3-5	Floating source and RSE input connections.....	3-7
	Figure 3-6	Ground-referenced sources and NRSE input connections.....	3-8
	Figure 3-7	Ground-referenced source and differential input .....	3-8
	Figure 3-8	Floating source and differential input.....	3-9
	Figure 3-9	Synchronous Digital Inputs Block Diagram.....	3-10
	Figure 3-10	Synchronous Digital Inputs timing .....	3-10
	Figure 3-11	Scan timing .....	3-13
	Figure 3-12	Pre-trigger (trigger occurs after M scans).....	3-15
	Figure 3-13	Pre-trigger (trigger with scan is in progress).....	3-15
	Figure 3-14	Pre-trigger with M_enable = 0 (trigger occurs before M scans)...	3-16
	Figure 3-15	Pre-trigger with M_enable = 1 .....	3-16
	Figure 3-16	Middle trigger with M_enable = 1.....	3-17
	Figure 3-17	Middle trigger (trigger occurs when a scan is in progress) .....	3-17
	Figure 3-18	Post trigger .....	3-18
	Figure 3-19	Delay trigger .....	3-18
	Figure 3-20	Post trigger with retrigger.....	3-19
	Figure 3-21	Scatter/gather DMA for data transfer.....	3-20
	Figure 3-22	Typical D/A timing of waveform generation .....	3-22
	Figure 3-23	Post trigger waveform generation.....	3-23
	Figure 3-24	Delay trigger waveform generation.....	3-23
	Figure 3-25	Re-triggered waveform generation with Post-trigger and DLY2_Counter = 0 .....	3-24
	Figure 3-26	Finite iterative waveform generation with Post-trigger and DLY2_Counter = 0 .....	3-24
	Figure 3-27	Infinite iterative waveform generation with Post-trigger and DLY2_Counter = 0 .....	3-25
	Figure 3-28	Stop mode I.....	3-26
	Figure 3-29	Stop mode II.....	3-26
	Figure 3-30	Stop mode III.....	3-26
	Figure 3-31	Mode 1 operation .....	3-28
	Figure 3-32	Mode 2 operation .....	3-28
	Figure 3-33	Mode 3 operation .....	3-28
	Figure 3-34	Mode 4 operation .....	3-29
	Figure 3-35	Mode 5 operation .....	3-29
	Figure 3-36	Mode 6 operation .....	3-29

<b>Section</b>	<b>Figure</b>	<b>Title</b>	<b>Page</b>
3	Figure 3-37	Mode 7 operation .....	3-30
	Figure 3-38	Mode 8 operation .....	3-30
	Figure 3-39	Analog trigger block diagram .....	3-31
	Figure 3-40	Below-Low analog trigger condition .....	3-31
	Figure 3-41	Above-High analog trigger condition .....	3-32
	Figure 3-42	Inside-Region analog trigger condition .....	3-32
	Figure 3-43	High-Hysteresis analog trigger condition .....	3-33
	Figure 3-44	Low-Hysteresis analog trigger condition .....	3-33
	Figure 3-45	External digital trigger .....	3-34
	Figure 3-46	DAQ signals routing .....	3-34
<b>Appendix</b>	<b>Figure</b>	<b>Title</b>	<b>Page</b>
A	Figure A-1	Open Project dialog box .....	A-3
	Figure A-2	Driver Configuration window .....	A-5
	Figure A-3	KDAQ-DRVR application building blocks .....	A-15
	Figure A-4	Typical function flow for all types of KDAQ-DRVR series .....	A-16
	Figure A-5	Fills channel gain queue first .....	A-17
	Figure A-6	Synchronous operation .....	A-18
	Figure A-7	Non-double buffered asynchronous operation .....	A-19
	Figure A-8	Double buffered asynchronous operation .....	A-20
	Figure A-9	All types of KPXI-DRVR series .....	A-22
	Figure A-10	Fills channel gain queue first .....	A-23
	Figure A-11	All types of KDAQ-DRVR series .....	A-25
	Figure A-12	Fills channel gain queue first .....	A-26
	Figure A-13	All types of KDAQ-DRVR series .....	A-27
	Figure A-14	Fills channel gain queue first .....	A-29
	Figure A-15	All types of KDAQ-DRVR series .....	A-31
	Figure A-16	Fills channel gain queue first .....	A-32
	Figure A-17	All types of KDAQ-DRVR series .....	A-33
	Figure A-18	Fills channel gain queue first .....	A-35
	Figure A-19	One-shot analog output programming .....	A-37
	Figure A-20	One-shot analog output programming .....	A-38
	Figure A-21	Non-double-buffered asynchronous continuous analog output programming .....	A-39
	Figure A-22	Non-double-buffered asynchronous continuous analog output programming .....	A-40
	Figure A-23	Double-buffered asynchronous continuous analog output programming .....	A-41
	Figure A-24	Double-buffered asynchronous continuous analog output programming .....	A-43
	Figure A-25	Typical flow of asynchronous analog output operation .....	A-44
	Figure A-26	Typical flow of asynchronous analog output operation .....	A-45
	Figure A-27	Typical flow of double-buffered asynchronous analog output operation .....	A-47
	Figure A-28	Typical flow of double-buffered asynchronous analog output operation .....	A-49
	Figure A-29	One-shot digital input programming .....	A-50
	Figure A-30	Typical flow of non-buffered single-point digital output operation .....	A-51
	Figure A-31	Double buffer mode principle .....	A-53

<b>Appendix Figure</b>	<b>Title</b>	<b>Page</b>
B	Figure B-1 Scan timing example .....	B-13
	Figure B-2 Scan timing example .....	B-15
	Figure B-3 Scan timing example .....	B-17
	Figure B-4 Scan timing example .....	B-20
	Figure B-5 Scan timing example .....	B-22
	Figure B-6 Scan timing example .....	B-25
	Figure B-7 Scan timing example .....	B-27
	Figure B-8 Scan timing example .....	B-30
	Figure B-9 DATA file format .....	B-98
	Figure B-10 DAQ File Conversion Utility .....	B-100
C	Figure C-1 Function Browser Options .....	C-2
	Figure C-2 Functions palette .....	C-3
	Figure C-3 Keithley PXI Devices Explorer .....	C-5
D	Figure D-1 Analog input palette .....	D-3
	Figure D-2 Analog output palette .....	D-21
	Figure D-3 Digital I/O palette .....	D-33

This page left blank intentionally.

# List of Tables

Section	Table	Title	Page
3	Table 3-1	Legend of 68-pin VHDCI-type connectors .....	3-5
	Table 3-2	Legend of SSI signals .....	3-6
	Table 3-3	Input ranges and output digital code (KPXI-DAQ-64-3M/ KPXI-DAQ-96-3M) .....	3-10
	Table 3-4	Unipolar analog input range (KPXI-DAQ-64-3M/ KPXI-DAQ-96-3M) .....	3-10
	Table 3-5	Bipolar analog input range (KPXI-DAQ-64-500K/ KPXI-DAQ-64-250K) .....	3-11
	Table 3-6	Unipolar analog input range (KPXI-DAQ-64-500K/ KPXI-DAQ-64-250K) .....	3-11
	Table 3-7	Bipolar output code table (Vref=10V if internal reference is selected) .....	3-20
	Table 3-8	Unipolar output code table (Vref=10V if internal reference is selected) .....	3-21
	Table 3-9	Analog trigger SRC1 (EXTATRIG) ideal transfer characteristic ...	3-31
	Table 3-10	Summary of user-controllable timing signals .....	3-34
	Table 3-11	Auxiliary function input signals .....	3-36
	Table 3-12	SSI timing signal summary .....	3-37
<b>Appendix</b>			
Table	Title	Page	
A	Table A-1	Initial default channel configuration .....	A-18
	Table A-2	Initial default AI configuration .....	A-18
	Table A-3	Initial default channel configuration .....	A-38
	Table A-4	Initial default DA configuration .....	A-38
B	Table B-1	Suggested data types .....	B-2
	Table B-2	Example trigger condition selection (KDAQ_AIO_Config) .....	B-46
	Table B-3	Status codes returned by KDAQ-DRVR .....	B-94
	Table B-4	Analog input range of digitizers .....	B-96
	Table B-5	Valid values for each model .....	B-96
	Table B-6	AI data format .....	B-97
	Table B-7	Data file header .....	B-98
	Table B-8	Data structure of ChannelRange unit (length: 2 bytes) .....	B-99
D	Table D-1	KI AI acquire waveform .....	D-3
	Table D-2	KI AI acquire waveforms .....	D-4
	Table D-3	KI AI sample channel .....	D-6
	Table D-4	KI AI sample channels .....	D-6
	Table D-5	KI AI clear .....	D-7
	Table D-6	KI AI config .....	D-9
	Table D-7	2-byte binary array .....	D-12
	Table D-8	Scaled and Binary Arrays .....	D-14
	Table D-9	Scaled Array .....	D-16
	Table D-10	KI AI single scan .....	D-17

<b>Appendix Table</b>	<b>Title</b>	<b>Page</b>
D	Table D-11 KI AI start .....	D-19
	Table D-12 KI AO generate waveform .....	D-22
	Table D-13 KI AO generate waveforms .....	D-22
	Table D-14 KI AO update channel .....	D-23
	Table D-15 KI AO update channels .....	D-24
	Table D-16 KI AO clear .....	D-25
	Table D-17 KI AO Config .....	D-25
	Table D-18 KI AO start .....	D-27
	Table D-19 KI AO wait .....	D-28
	Table D-20 KI AO write binary array .....	D-29
	Table D-21 KI AO write binary array scaled array .....	D-30
	Table D-22 KI AO Trigger and Gate Config .....	D-32
	Table D-23 KI Read from Digital Line .....	D-34
	Table D-24 KI Read from Digital Port .....	D-34
	Table D-25 KI Write to Digital Line .....	D-35
	Table D-26 KI Write to Digital Port .....	D-36
	Table D-27 KI DIO Clear .....	D-37
	Table D-28 KI DIO Config .....	D-38
	Table D-29 KI DIO Read .....	D-40
	Table D-30 KI DIO Start .....	D-42
	Table D-31 KI DIO Write .....	D-43
	Table D-32 KI DIO Port Config .....	D-45
	Table D-33 KI Count Events or Time .....	D-46
	Table D-34 KI Generate Delayed Pulse .....	D-47
	Table D-35 KI Generate Pulse-Train .....	D-48
	Table D-36 KI Measure Pulse-Width or Period .....	D-49
	Table D-37 KI Continuous Pulse Generator Config .....	D-50
	Table D-38 KI Counter Divider Config .....	D-52
	Table D-39 KI Counter Read .....	D-53
	Table D-40 KI Counter Start .....	D-54
	Table D-41 KI Counter Stop .....	D-55
	Table D-42 KI Delayed Pulse Generator Config .....	D-56
	Table D-43 KI Down Counter or Divider Config .....	D-58
	Table D-44 KI Event or Time Counter Config .....	D-59
	Table D-45 KI Pulse-Width or Period Measurement Config .....	D-61
	Table D-46 KI UpDown Counter Config .....	D-62
	Table D-47 KI ICTR Control .....	D-63
	Table D-48 KI KPXI-DAQ series devices and Digitizer Series Calibrate .....	D-67
	Table D-49 KI Route Signal .....	D-68
	Table D-50 KI SSI Control .....	D-69
	Table D-51 KI Error Handler .....	D-70
	Table D-52 Error Codes: KIDAQ LabVIEW VIs .....	D-71
	Table D-53 Analog Input Range .....	D-73
	Table D-54 Valid analog input ranges (specified by module) .....	D-75
	Table D-55 Analog Input data format (by Model) .....	D-76



**In this section:**

Topic	Page
<b>Introduction</b> .....	1-2
Features .....	1-2
Applications .....	1-2
<b>Safety symbols and terms</b> .....	1-2
<b>Specifications</b> .....	1-3
<b>Unpacking and inspection</b> .....	1-3
Inspection for damage .....	1-3
Shipment contents .....	1-3
Instruction manual .....	1-4
Repacking for shipment .....	1-4
<b>Software introduction</b> .....	1-4
Programming library KDAQ-DRVR .....	1-4
KDAQ-LVIEW LabVIEW® driver .....	1-4

## Introduction

The KPXI series modules are advanced data acquisition cards based on the 32-bit PCI architecture. High performance designs and the state-of-the-art technology make this card ideal for data logging and signal analysis applications in many production test systems.

This manual is designed to help you use/understand the Model KPXI-DAQ module. It describes the versatile functions and the operation theory of the Model KPXI-DAQ module.

## Features

KPXI-DAQ module Advanced Data Acquisition Cards provide the following advanced features:


- 32-bit PCI-Bus, plug and play
- Up to 96 single-ended inputs or 48 differential inputs, mixing of SE and DI analog input signals are possible
- Up to 1024 words analog input Channel Gain Queue configuration size
- KPXI-DAQ-64-3M/KPXI-DAQ-96-3M: 12-bit Analog input resolution with sampling rate up to 3MHz
- KPXI-DAQ-64-500K: 16-bit Analog input resolution with sampling rate up to 500KHz
- KPXI-DAQ-64-250K: 16-bit Analog input resolution with sampling rate up to 250KHz
- Programmable Bipolar/Unipolar analog input
- Programmable gain:  
KPXI-DAQ-64-3M/KPXI-DAQ-96-3M: x1, x2, x4, x5, x8, x10, x20, x40, x50, x200.  
KPXI-DAQ-64-500K/KPXI-DAQ-64-250K: x1, x2, x4, x8.
- A/D FIFO size: 1024 samples
- Versatile trigger sources: software trigger, external digital trigger, analog trigger and trigger from System Synchronization Interface (SSI)
- A/D Data transfer: software polling & bus-mastering DMA with Scatter/Gather functionality
- Four A/D trigger modes: post-trigger, delay-trigger, pre-trigger and middle-trigger
- 2 channel D/A outputs with waveform generation capability (KPXI-DAQ-96-3M doesn't provide this function)
- 1024 word length output data FIFO for D/A channels
- D/A Data transfer: software update and bus-mastering DMA with Scatter/Gather functionality
- System Synchronization Interface (SSI)
- A/D and D/A fully auto-calibration
- Completely jumper-less and software configurable


## Applications


- Automotive Testing
- Transient signal measurement
- ATE

## Safety symbols and terms

The following symbols and terms may be found on the Model KPXI-DAQ module or used in this manual.

The  symbol indicates that the user should refer to the operating instructions located in the manual.

The  symbol shows that high voltage may be present on the terminal(s). Use standard safety precautions to avoid personal contact with these voltages.

The  symbol on an instrument shows that the surface may be hot. Avoid personal contact to prevent burns.

The **WARNING** heading used in this manual explains dangers that might result in personal injury or death. Always read the associated information very carefully before performing the indicated procedure.

The **CAUTION** heading used in this manual explains hazards that could damage the unit. Such damage may invalidate the warranty.

## Specifications

Refer to the product data sheet for updated KIDAQ® KPXI Multi-Function PXI Module's specifications. Check the Keithley Instruments website at [www.keithley.com](http://www.keithley.com) for the latest updates to the specifications.

## Unpacking and inspection

### Inspection for damage

**CAUTION** Your KPXI-DAQ Series module contains electro-static sensitive components that can easily be damaged by static electricity.

**Therefore, handle the card on a grounded anti-static mat. The operator should be wearing an anti-static wristband, grounded at the same point as the anti-static mat.**

The Model KPXI-DAQ Series Module was carefully inspected electrically and mechanically before shipment.

Inspect the card module carton for obvious damages. Shipping and handling may damage the module. Make sure there are no shipping and handling damages on the module's carton before continuing.

After opening the card module carton, extract the module and place it only on a grounded anti-static surface with component side up. Save the original packing carton for possible future shipment.

Again, inspect the module for damages. Report any damage to the shipping agent immediately.

### Shipment contents

The following items are included with every Model KPXI-DAQ Series order:

- Model KPXI-DAQ series module
- CD containing required software and manuals

## Instruction manual

A CD-ROM containing this User's Manual and required software is included with each Model KPXI-DAQ series order. If a hardcopy of the Model KPXI-DAQ Series User's Manual is required, you can order the Manual Package (Keithley Instruments Part Number KPXI-DAQ-901-01). The Manual Package includes an instruction manual and any pertinent addenda.

Always check the Keithley Instruments website at [www.keithley.com](http://www.keithley.com) for the latest revision of the manual. The latest manual can be downloaded (in PDF format) from the website.

## Repacking for shipment

**CAUTION** The boards must be protected from static discharge and physical shock. Never remove any of the socketed parts except at a static-free workstation. Use the anti-static bag shipped with the product to handle the board. Wear a grounded wrist strap when servicing.

Should it become necessary to return the Model KPXI-DAQ Series module for repair, carefully pack the unit in its original packing carton or the equivalent, and follow these instructions:

- Call Keithley Instruments' repair department at 1-888-KEITHLEY (1-888-534-8453) for a Return Material Authorization (RMA) number.
- Let the repair department know the warranty status of the Model KPXI-DAQ module.
- Write ATTENTION REPAIR DEPARTMENT and the RMA number on the shipping label.
- Complete and include the Service Form located at the back of this manual.

## Software introduction

This section contains information on provided software. Keithley Instruments' provides versatile software drivers and packages for different systems. Keithley Instruments not only provides programming libraries such as DLL's for most Windows® based systems, but also drivers for other software packages such as LabVIEW.<sup>1</sup>

All software options are included in the Keithley Instruments' CD.

## Programming library KDAQ-DRVR

KDAQ-DRVR includes device drivers and DLL's for Windows XP® and Windows 2000®. Therefore, all applications developed with KDAQ-DRVR are compatible on Windows XP/2000. The developing environment can be VB, VC++, BC5, or any Windows programming language that allows calls to a DLL. Documentation includes a User's Guide (refer to [Appendix A: KDAQ-DRVR User's Guide](#)), and a Function Reference (refer to [Appendix B: KDAQ-DRVR Function Reference](#)).

## KDAQ-LVIEW LabVIEW® driver

KDAQ-LVIEW contains the VI's, which are used to interface with National Instrument's® Lab-VIEW® software package. The KDAQ-LVIEW supports Windows XP/2000. The LabVIEW driver is shipped free with the board. Documentation includes an Interface Guide (refer to [Appendix C: KIDAQ®-LabVIEW Compatible Interface Guide](#)), and an interface Function Reference (refer to [Appendix D: KIDAQ®-LabVIEW Compatible Function Reference](#)).

---

1. National Instruments™, NI, and LabVIEW are trademarks of the National Instruments Corporation.

**In this section:**

<b>Topic</b>	<b>Page</b>
Introduction .....	2-2
Handling precautions .....	2-2
Mechanical Drawing.....	2-2
Configuration.....	2-2
Plug and Play .....	2-2
Configuration .....	2-3
Troubleshooting.....	2-3
Installation .....	2-3

## Introduction

This section contains information about handling and installing Keithley Instruments KIDAQ® KPXI series cards:

- [Handling precautions](#)
- [Mechanical Drawing](#)
- [Configuration](#)
- [Installation](#)

## Handling precautions

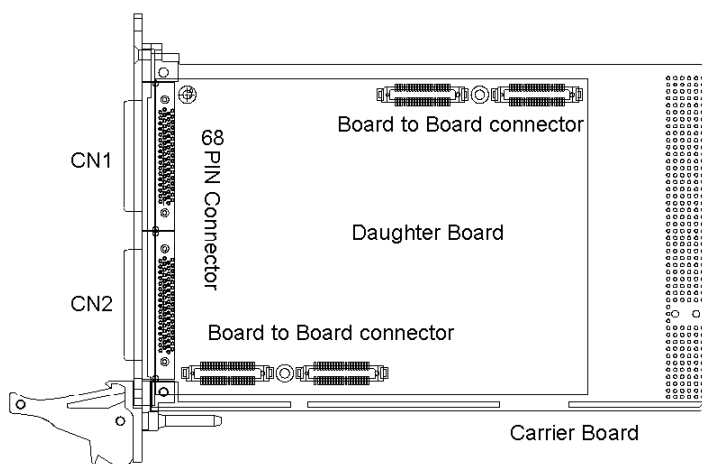
**CAUTION** Use care when handling the KIDAQ® KPXI series cards. KIDAQ® KPXI series cards contain electro-static sensitive components that can be easily damaged by static electricity.

When handling, make sure to observe the following guidelines:

- Only handle the card on a grounded anti-static mat.
- Wear an anti-static wristband that is grounded at the same point as the anti-static mat.

## Mechanical Drawing

Figure 2-1  
PXI Layout of KPXI-DAQ



## Configuration

### Plug and Play

As a plug and play component, the board requests an interrupt number via its PCI controller. The system BIOS responds with an interrupt assignment based on the board information and system parameters. These system parameters are determined by the installed drivers and the hardware load recognized by the system. If this is the first time a KIDAQ® KPXI series card will be installed on your Windows® system, a hardware driver needs to be installed. Refer to [Installation](#) for detailed information.

## Configuration

Configuration is done on a board-by-board basis for all PXI boards on your system. Configuration is controlled by the system and software. There is no jumper setting required (or available) for base address, DMA, and interrupt IRQ.

The configuration is not static, but is subject to change with every boot of the system as new boards are added or removed.

## Troubleshooting

If your system doesn't boot or if you experience erratic operation with your PXI board in place, it's likely caused by an interrupt conflict (perhaps the BIOS Setup is incorrectly configured). In general, the solution, is to consult the BIOS documentation that comes with your system.

## Installation

### Step 1. Install driver software

Windows® will find the new module automatically. If this is the first time a KPXI Series card is running on your Windows system, you will need to install a hardware driver. Use the following installation procedure as a guide.

**NOTE:** Keithley Instruments controllers are pre-loaded with the necessary drivers.

For Windows 2000/XP:

1. Insert the CD shipped with the module. The CD should auto load. From the base menu install the KDAQ-DRVR. This is the hardware driver that recognizes the KPXI Series modules. If the CD does not auto load run, then under `x:\KDAQ-DRVR\DISK1\`, you will find `SETUP.EXE` (x is the drive letter of your CDROM). This will also run the install.
2. When you complete driver installation, turn off the system.

### Step 2. Inspect module

Keeping the “[Handling precautions](#)” information in mind, inspect the module for damage. With the module placed on a firm, flat surface, press down on all socketed IC's to make sure that they are properly seated.

If the module does not pass the inspection, do not proceed with the installation.

**CAUTION** Do not apply power to the card if it has been damaged.

The KPXI Series card is now ready for installation.

### Step 3. Install module

Remove power from the system and install the KPXI card in an available slot.

The PXI connectors are rigid and require careful handling when inserted and removed. Improper handling of modules can easily damage the backplane.

To insert the module into a PXI chassis, use the following procedure as a guide (refer to [Figure 2-2](#)):

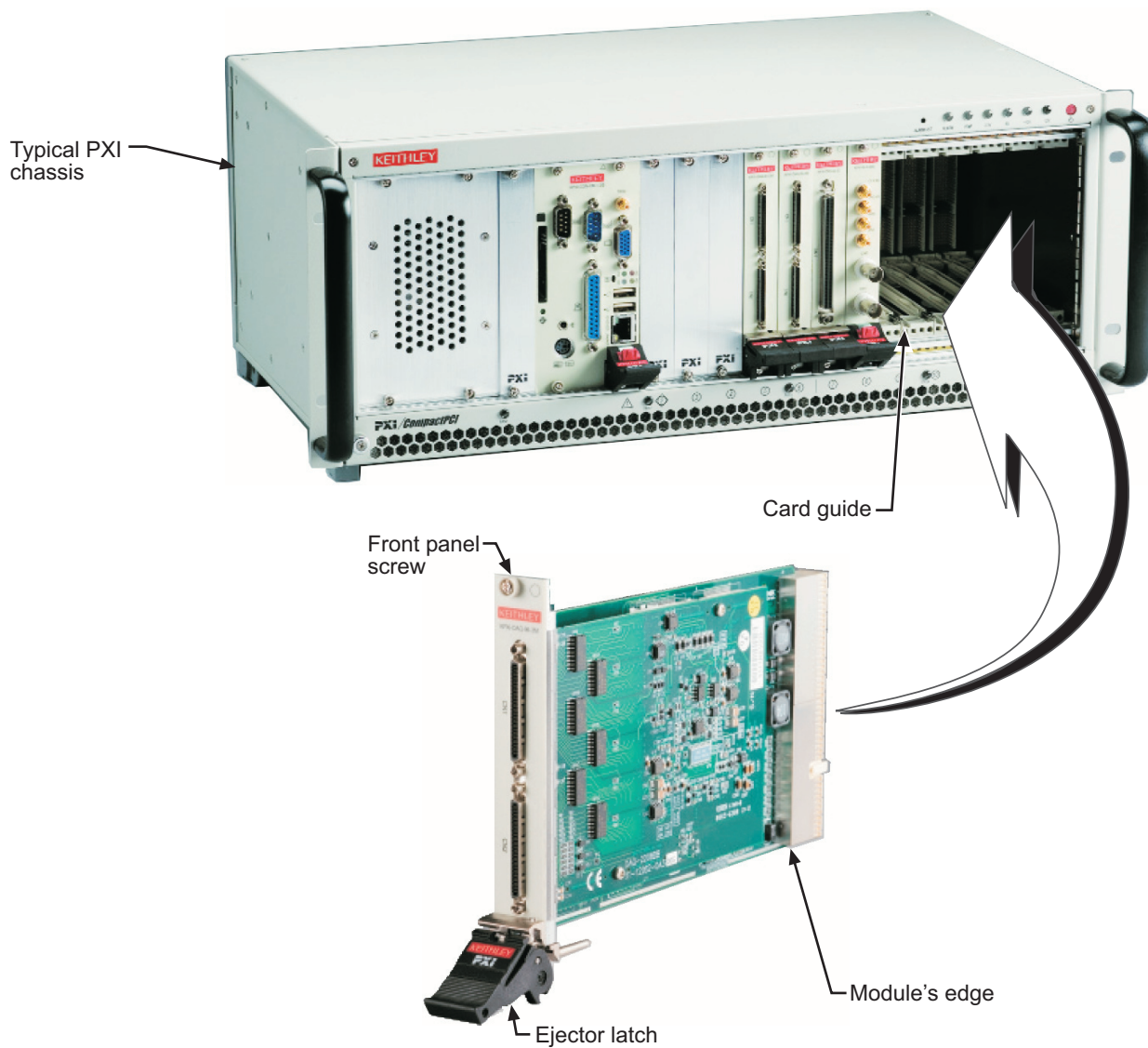
1. Turn off the system.
2. Align the module's edge with the card guide in the PXI chassis.

3. Slide the module into the chassis until resistance is felt from the PXI connector.
4. Push the ejector upwards and fully insert the module into the chassis. Once inserted, a "click" can be heard from the ejector latch.
5. Tighten the screw on the front panel.
6. Turn on the system.

To remove a module from a PXI chassis, use the following procedure as a guide:

1. Turn off the system.
2. Loosen the screw on the front panel.
3. Push the ejector downwards and carefully remove the module from the chassis.

Figure 2-2  
Typical PXI module installation



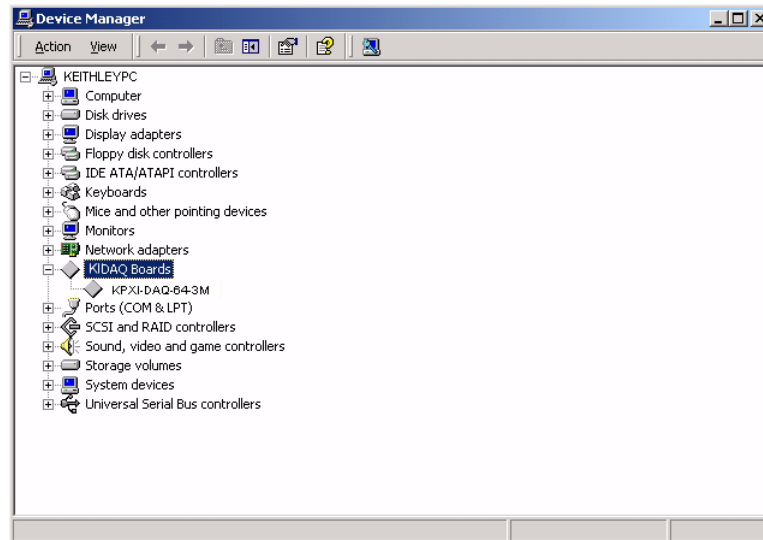


### Step 4. Verify installation

When the system is turned on for the first time with a new module present (or a module in a new slot), Windows **Add New Hardware Wizard** attempts to locate the correct driver. If it cannot find the correct driver, even after you have loaded the driver above in Step 1, then force the **Add New Hardware Wizard** to look in Windows system32 directory. The driver files should be in this location. If they are not, shutdown the system, remove the module, and restart the installation process.

When the **Add New Hardware Wizard** finishes, the window will verify whether or not installation was successful. To confirm if the module is installed correctly at a later time, use **Windows Device Manager**. In the **Device Manager** under KIDAQ Boards, look for a device name matching the model number of the newly installed board (see [Figure 2-3](#) for an example). If it is found, installation is complete. If the board appears with an exclamation point or warning in Device Manager, the installation was unsuccessful. If unsuccessful, use **Device Manager** to update the driver or un-install the module, power down the system, remove the module, and attempt installation again from Step 1.

Figure 2-3  
**Device manager (successful installation)**



This page left blank intentionally.

**In this section:**

<b>Topic</b>	<b>Page</b>
Introduction .....	3-2
Signal Connections .....	3-2
Connectors Pin Assignment .....	3-2
Analog Input Signal Connection .....	3-7
Operation Theory .....	3-9
A/D Conversion .....	3-9
D/A Conversion .....	3-20
Digital I/O .....	3-26
General Purpose Timer/Counter Operation .....	3-27
Trigger Sources .....	3-30
User-controllable Timing Signals .....	3-34
Calibration .....	3-38
Loading Calibration Constants .....	3-38
Auto-calibration .....	3-38
Saving Calibration Constants .....	3-39

## Introduction

This section contains operation information on KIDAQ® KPXI series cards including signal connections. Use this information to aid in the understanding of how to configure and program the KIDAQ® KPXI series modules.

## Signal Connections

This section describes the connectors of the KPXI-DAQ module, and the signal connection between the KPXI-DAQ module and external devices.

### Connectors Pin Assignment

Each KPXI-DAQ module is equipped with two 68-pin VHDCI-type connectors (AMP-787254-1). They are used for digital input / output, analog input / output, and timer/counter signaling, etc. The pin assignment for the connectors are illustrated in [Figure 3-1](#), [Figure 3-2](#), [Figure 3-3](#), [Figure 3-4](#), and also [Table 3-1](#).

Figure 3-1

#### CN1 pinout (KPXI-DAQ-64-3M/KPXI-DAQ-64-500K/KPXI-DAQ-64-250K)

AI0 (AIH0)	1	35	(AIL0)	AI32
AI1 (AIH1)	2	36	(AIL1)	AI33
AI2 (AIH2)	3	37	(AIL2)	AI34
AI3 (AIH3)	4	38	(AIL3)	AI35
AI4 (AIH4)	5	39	(AIL4)	AI36
AI5 (AIH5)	6	40	(AIL5)	AI37
AI6 (AIH6)	7	41	(AIL6)	AI38
AI7 (AIH7)	8	42	(AIL7)	AI39
AI8 (AIH8)	9	43	(AIL8)	AI40
AI9 (AIH9)	10	44	(AIL9)	AI41
AI10 (AIH10)	11	45	(AIL10)	AI42
AI11 (AIH11)	12	46	(AIL11)	AI43
AI12 (AIH12)	13	47	(AIL12)	AI44
AI13 (AIH13)	14	48	(AIL13)	AI45
AI14 (AIH14)	15	49	(AIL14)	AI46
AI15 (AIH15)	16	50	(AIL15)	AI47
AISENSE	17	51	AI GND	
AI16 (AIH16)	18	52	(AIL16)	AI48
AI17 (AIH17)	19	53	(AIL17)	AI49
AI18 (AIH18)	20	54	(AIL18)	AI50
AI19 (AIH19)	21	55	(AIL19)	AI51
AI20 (AIH20)	22	56	(AIL20)	AI52
AI21 (AIH21)	23	57	(AIL21)	AI53
AI22 (AIH22)	24	58	(AIL22)	AI54
AI23 (AIH23)	25	59	(AIL23)	AI55
AI24 (AIH24)	26	60	(AIL24)	AI56
AI25 (AIH25)	27	61	(AIL25)	AI57
AI26 (AIH26)	28	62	(AIL26)	AI58
AI27 (AIH27)	29	63	(AIL27)	AI59
AI28 (AIH28)	30	64	(AIL28)	AI60
AI29 (AIH29)	31	65	(AIL29)	AI61
AI30 (AIH30)	32	66	(AIL30)	AI62
AI31 (AIH31)	33	67	(AIL31)	AI63
EXTATRIG	34	68	AI GND	

\* Symbols in “( )” are for differential mode connection.

Figure 3-2  
**CN2 pinout (KPXI-DAQ-64-3M/KPXI-DAQ-64-500K/KPXI-DAQ-64-250K)**

DA0OUT	1	35	AOGND
DA1OUT	2	36	AOGND
AOEXTREF	3	37	AOGND
NC	4	38	NC
DGND	5	39	DGND
EXTWFTRIG	6	40	DGND
EXTDTRIG	7	41	DGND
SSHOUT	8	42	SDI0 / DGND*
RESERVED	9	43	SDI1 / DGND*
RESERVED	10	44	SDI2 / DGND*
AFI1	11	45	SDI3 / DGND*
AFI0	12	46	DGND
GPTC0_SRC	13	47	DGND
GPTC0_GATE	14	48	DGND
GPTC0_UPDOWN	15	49	DGND
GPTC0_OUT	16	50	DGND
GPTC1_SRC	17	51	DGND
GPTC1_GATE	18	52	DGND
GPTC1_UPDOWN	19	53	DGND
GPTC1_OUT	20	54	DGND
EXTTIMEBASE	21	55	DGND
PB7	22	56	PB6
PB5	23	57	PB4
PB3	24	58	PB2
PB1	25	59	PB0
PC7	26	60	PC6
PC5	27	61	PC4
DGND	28	62	DGND
PC3	29	63	PC2
PC1	30	64	PC0
PA7	31	65	PA6
PA5	32	66	PA4
PA3	33	67	PA2
PA1	34	68	PA0

\*Pin 42~45 are SDI<0..3> for KPXI-DAQ-64-3M; DGND for KPXI-DAQ-64-500K/  
 KPXI-DAQ-64-250K

Figure 3-3  
**CN1 pinout (KPXI-DAQ-96-3M)**

AI0 (AIH0)	1	35	(AIL0)	AI48
AI1 (AIH1)	2	36	(AIL1)	AI49
AI2 (AIH2)	3	37	(AIL2)	AI50
AI3 (AIH3)	4	38	(AIL3)	AI51
AI4 (AIH4)	5	39	(AIL4)	AI52
AI5 (AIH5)	6	40	(AIL5)	AI53
AI6 (AIH6)	7	41	(AIL6)	AI54
AI7 (AIH7)	8	42	(AIL7)	AI55
AISENSE	9	43	AIGND	
AI8 (AIH8)	10	44	(AIL8)	AI56
AI9 (AIH9)	11	45	(AIL9)	AI57
AI10 (AIH10)	12	46	(AIL10)	AI58
AI11 (AIH11)	13	47	(AIL11)	AI59
AI12 (AIH12)	14	48	(AIL12)	AI60
AI13 (AIH13)	15	49	(AIL13)	AI61
AI14 (AIH14)	16	50	(AIL14)	AI62
AI15 (AIH15)	17	51	(AIL15)	AI63
AI16 (AIH16)	18	52	(AIL16)	AI64
AI17 (AIH17)	19	53	(AIL17)	AI65
AI18 (AIH18)	20	54	(AIL18)	AI66
AI19 (AIH19)	21	55	(AIL19)	AI67
AI20 (AIH20)	22	56	(AIL20)	AI68
AI21 (AIH21)	23	57	(AIL21)	AI69
AI22 (AIH22)	24	58	(AIL22)	AI70
AI23 (AIH23)	25	59	(AIL23)	AI71
AIGND	26	60	AIGND	
AI24 (AIH24)	27	61	(AIL24)	AI72
AI25 (AIH25)	28	62	(AIL25)	AI73
AI26 (AIH26)	29	63	(AIL26)	AI74
AI27 (AIH27)	30	64	(AIL27)	AI75
AI28 (AIH28)	31	65	(AIL28)	AI76
AI29 (AIH29)	32	66	(AIL29)	AI77
AI30 (AIH30)	33	67	(AIL30)	AI78
AI31 (AIH31)	34	68	(AIL31)	AI79

\* Symbols in “()” are for differential mode connection.

Figure 3-4  
**CN2 pinout (KPXI-DAQ-96-3M)**

AI32 (AIH32)	1	35	(AIL32)	AI80
AI33 (AIH33)	2	36	(AIL33)	AI81
AI34 (AIH34)	3	37	(AIL34)	AI82
AI35 (AIH35)	4	38	(AIL35)	AI83
AI36 (AIH36)	5	39	(AIL36)	AI84
AI37 (AIH37)	6	40	(AIL37)	AI85
AI38 (AIH38)	7	41	(AIL38)	AI86
AI39 (AIH39)	8	42	(AIL39)	AI87
EXTATRIG	9	43		AIGND
AI40 (AIH40)	10	44	(AIL40)	AI88
AI41 (AIH41)	11	45	(AIL41)	AI89
AI42 (AIH42)	12	46	(AIL42)	AI90
AI43 (AIH43)	13	47	(AIL43)	AI91
AI44 (AIH44)	14	48	(AIL44)	AI92
AI45 (AIH45)	15	49	(AIL45)	AI93
AI46 (AIH46)	16	50	(AIL46)	AI94
AI47 (AIH47)	17	51	(AIL47)	AI95
AIGND	18	52		AIGND
NC	19	53		NC
EXTDTRIG	20	54		AF10
EXTTIMEBASE	21	55		DGND
PB7	22	56		PB6
PB5	23	57		PB4
PB3	24	58		PB2
PB1	25	59		PB0
PC7	26	60		PC6
PC5	27	61		PC4
DGND	28	62		DGND
PC3	29	63		PC2
PC1	30	64		PC0
PA7	31	65		PA6
PA5	32	66		PA4
PA3	33	67		PA2
PA1	34	68		PA0

Table 3-1  
**Legend of 68-pin VHDCI-type connectors**

Signal Name	Reference	Direction	Description
AIGND	-----	-----	Analog ground for AI. All three ground references (AIGND, AOGND, and DGND) are connected together on board
AI<0..63/95>	AIGND	Input	*For KPXI-DAQ-64-3M/KPXI-DAQ-64-500K/KPXI-DAQ-64-250K Analog Input Channels 0~63. Each channel pair, AI<i, i+32> (i=0..31) can be configured either two single-ended inputs or one differential input pair (marked as AIH<0..31> and AIL<0..31>) *For KPXI-DAQ-96-3M only: Analog Input Channels 0~95. Each channel pair, AI<i, i+48> (i=0..47) can be configured either two single-ended inputs or one differential input pair (marked as AIH<0..47> and AIL<0..47>)
AISENSE	AIGND	Input	Analog Input Sense. This pin is the reference for any channels AI<0..63> in NRSE input configuration
EXTATRIG	AIGND	Input	External AI analog trigger

Table 3-1 (continued)  
Legend of 68-pin VHDCI-type connectors

Signal Name	Reference	Direction	Description
DA0OUT	AOGND	Output	AO channel 0
DA1OUT	AOGND	Output	AO channel 1
AOEXTREF	AOGND	Input	External reference for AO channels
AOGND	-----	-----	Analog ground for AO
EXTWFTRIG	DGND	Input	External AO waveform trigger
EXTDTRIG	DGND	Input	External AI digital trigger
RESERVED	-----	Output	Reserved. Please leave it open
SDI<0..3> (for KPXI-DAQ-64-3M only)	DGND	Input	Synchronous digital inputs. These 4 digital inputs are sampled simultaneously with the analog signal input
GPTC<0,1>_SRC	DGND	Input	Source of GPTC<0,1>
GPTC<0,1>_GATE	DGND	Input	Gate of GPTC<0,1>
GPTC<0,1>_OUT	DGND	Input	Output of GPTC<0,1>
GPTC<0,1>_UPDOWN	DGND	Input	Up/Down of GPTC<0,1>
EXTTIMEBASE	DGND	Input	External Timebase
DGND	-----	-----	Digital ground
PB<7,0>	DGND	PIO*	Programmable DIO of 8255 Port B
PC<7,0>	DGND	PIO*	Programmable DIO of 8255 Port C
PA<7,0>	DGND	PIO*	Programmable DIO of 8255 Port A
AFI0	DGND	Input	Auxiliary Function Input 0 (ADCONV, AD_START)
AFI1	DGND	Input	Auxiliary Function Input 1 (DAWR, DA_START)

Table 3-2  
Legend of SSI signals

**NOTE** The System Synchronization Interface (SSI) signals can be routed to the PXI trigger bus for multiple module synchronization within a chassis.

SSI timing signal	Functionality
SSI_TIMEBASE	SSI master: send the TIMEBASE out SSI slave: accept the SSI_TIMEBASE to replace the internal TIMEBASE signal.
SSI_ADCONV	SSI master: send the ADCONV out SSI slave: accept the SSI_ADCONV to replace the internal ADCONV signal.
SSI_SCAN_START	SSI master: send the SCAN_START out SSI slave: accept the SSI_SCAN_START to replace the internal SCAN_START signal.
SSI_AD_TRIG	SSI master: send the internal AD_TRIG out SSI slave: accept the SSI_AD_TRIG as the digital trigger signal.
SSI_DAWR	SSI master: send the DAWR out. SSI slave: accept the SSI_DAWR to replace the internal DAWR signal.
SSI_DA_TRIG	SSI master: send the DA_TRIG out. SSI slave: accept the SSI_DA_TRIG as the digital trigger signal.



## Analog Input Signal Connection

The KPXI-DAQ module provides up to 96 single-ended or 48 differential analog input channels. You can fill the Channel Gain Queue to get desired combination of the input signal types. The analog signal can be converted to digital value by the A/D converter. To avoid ground loops and obtain a more accurate measurement from the A/D conversion, it is quite important to understand the signal source type and how to choose the analog input modes: RSE, NRSE, and DIFF mode.

### Types of signal sources

#### Floating Signal Sources

A floating signal source means it is not connected in any way to the building's ground system. A device with an isolated output is a floating signal source, such as optical isolator outputs, transformer outputs, and thermocouples.

#### Ground-Referenced Signal Sources

A ground-referenced signal means it is connected in some way to the building's system. That is, the signal source is already connected to a common ground point with respect to the KPXI-DAQ module, assuming that the computer is plugged into the same power system. Non-isolated outputs of instruments and devices that plug into the building's power system are ground-referenced signal sources.

### Input Configurations

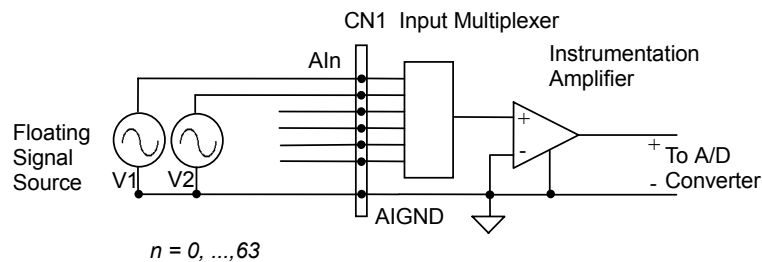
#### Single-ended Connections

A single-ended connection is used when the analog input signal is referenced to a ground that can be shared with other analog input signals. There are 2 different types for single-ended connections: RSE and NRSE configuration. In RSE configuration, the KPXI-DAQ module board provides the grounding point for the external analog input signals and is suitable for floating signal sources. While in NRSE configuration the board doesn't provide the grounding point, the external analog input signal provides its own reference grounding point and is suitable for ground-referenced signals.

#### Referenced Single-ended (RSE) Mode

In referenced single-ended mode, all the input signals are connected to the ground provided by the KPXI-DAQ module. It is suitable for connections with floating signal sources. Figure 3-5 shows an illustration. Note that when more than two floating sources are connected, these sources will be referenced to the same common ground.

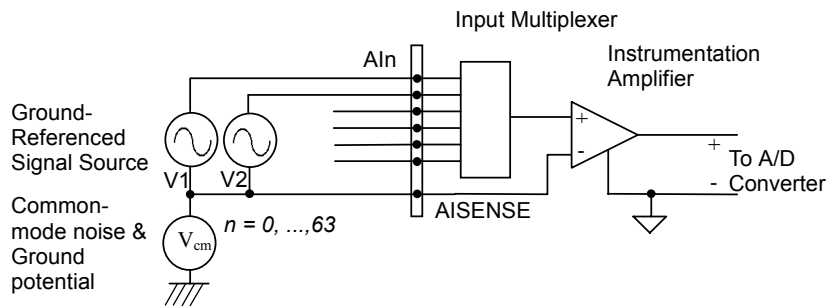
Figure 3-5  
**Floating source and RSE input connections**



### Non-Referenced Single-ended (NRSE) Mode

To measure ground-referenced signal sources, which are connected to the same ground point, you can connect the signals in NRSE mode. Figure 3-6 illustrates the connection. The signals local ground reference is connected to the negative input of the instrumentation Amplifier (AISENSE pin on CN1 connector), and the common-mode ground potential between signal ground and the ground on board will be rejected by the instrumentation amplifier.

Figure 3-6  
Ground-referenced sources and NRSE input connections



### Differential input mode

The differential input mode provides two inputs that respond to signal voltage difference between them. If the signal source is ground-referenced, the differential mode can be used for the common-mode noise rejection. Figure 3-7 shows the connection of ground-referenced signal sources under differential input mode.

Figure 3-7  
Ground-referenced source and differential input

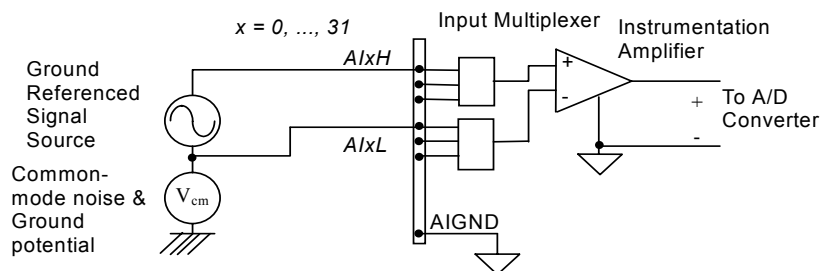
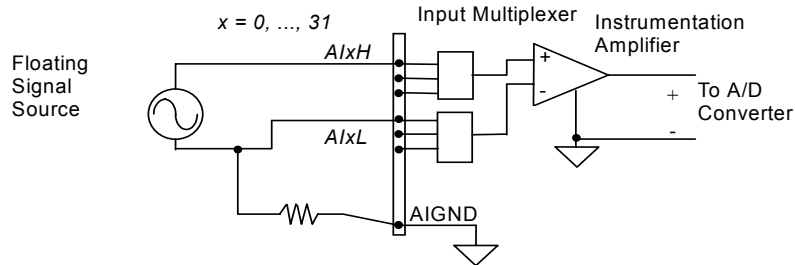


Figure 3-8 shows how to connect a floating signal source to the KPXI-DAQ module in differential input mode. For floating signal sources, you need to add a resistor at each channel to provide a bias return path. The resistor value should be about 100 times the equivalent source impedance. If the source impedance is less than 100ohms, you can simply connect the negative side of the signal to AIGND as well as the negative input of the Instrumentation Amplifier without any resistors. In differential input mode, less noise couples into the signal connections than in single-ended mode.

Figure 3-8  
**Floating source and differential input**



## Operation Theory

The operational theory of KPXI-DAQ module functions are described in this section. The functions include the A/D conversion, D/A conversion, Digital I/O and General Purpose Counter / Timer. The operation theory can help you understand how to configure and program the KPXI-DAQ module.

### A/D Conversion

When using an A/D converter, users should first know about the properties of the signal to be measured. Users can decide which channel to use and where to connect the signals to the card. For more information, refer to [Analog Input Signal Connection](#). In addition, users should define and control the A/D signal configurations, including channels, gains, and polarities (Unipolar/ Bipolar).

The A/D acquisition is initiated by a trigger source; users must decide how to trigger the A/D conversion. The data acquisition will start once a trigger condition is matched.

After the end of an A/D conversion, the A/D data is buffered in a Data FIFO. The A/D data can now be transferred into the PC's memory for further processing.

Two acquisition modes, Software Polling and Scan acquisition are described below. Timing, trigger modes, trigger sources, and transfer methods are included.

### KPXI-DAQ-64-3M/KPXI-DAQ-96-3M AI Data Format

#### Synchronous Digital Inputs (for KPXI-DAQ-64-3M only)

When each AD conversion is completed, the 12-bit converted digital data accompanied with 4 bits of SDI<3..0> from CN2 will be latched into the 16-bit register and data FIFO, as shown in [Figure 3-9](#) and [Figure 3-10](#). Therefore, users can simultaneously sample one analog signal with four digital signals. The data format of every acquired 16-bit data is of the form:

D11, D10, D9 ..... D1, D0, b3, b2, b1, b0

Where:

D11, D10, D9 ..... D1, D0: 2's complement A/D 12-bit data

b3, b2, b1, b0: Synchronous Digital Inputs SDI<3..0>

Figure 3-9  
Synchronous Digital Inputs Block Diagram

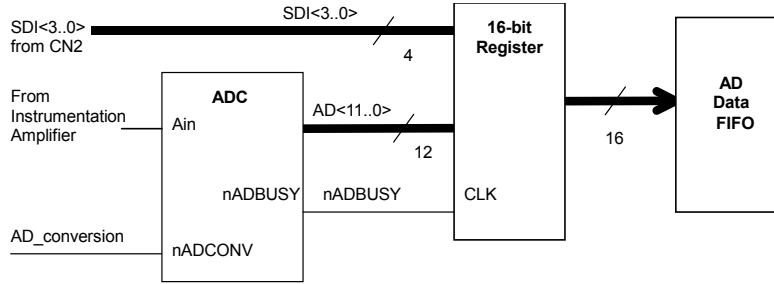
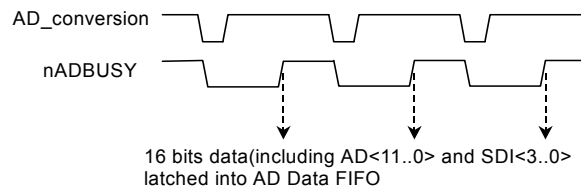


Figure 3-10  
Synchronous Digital Inputs timing



**NOTE** The analog signal is sampled when an AD conversion starts (falling edge of signal AD\_conversion), while SDI<3..0> are sampled right after an AD conversion is completed (rising edge of signal nADBUSHY). To be precise: SDI<3..0> are sampled with 280ns lag to the analog signal.

Table 3-3 and Table 3-4 illustrate the ideal transfer characteristics of some input ranges.

Table 3-3  
Input ranges and output digital code (KPXI-DAQ-64-3M/KPXI-DAQ-96-3M)

**NOTE** The last 4 digital codes are SDI<3..0> and is supported in KPXI-DAQ-64-3M only.

Description	Bipolar Analog Input Range				Digital Code
	±10V	±5V	±2.5V	±1.25V	
Full-scale Range	±10V	±5V	±2.5V	±1.25V	
Least significant bit	4.88mV	2.44mV	1.22mV	0.61mV	
FSR-1LSB	9.9951V	4.9976V	2.4988V	1.2494V	7FFX
Midscale +1LSB	4.88mV	2.44mV	1.22mV	0.61mV	001X
Midscale	0V	0V	0V	0V	000X
Midscale -1LSB	-4.88mV	-2.44mV	-1.22mV	-0.61mV	FFFX
-FSR	-10V	-5V	-2.5V	-1.25V	800X

Table 3-4  
Unipolar analog input range (KPXI-DAQ-64-3M/KPXI-DAQ-96-3M)

**NOTE** The last 4 digital codes are SDI<3..0> and are supported in KPXI-DAQ-64-3M only.

Description	Unipolar Analog Input Range			Digital code
	0V to 10V	0 to +5V	0 to +2.5V	
Full-scale Range	0V to 10V	0 to +5V	0 to +2.5V	
Least significant bit	2.44mV	1.22mV	0.61mV	
FSR-1LSB	9.9976V	4.9988V	2.9994V	7FFX

Table 3-4 (continued)  
**Unipolar analog input range (KPXI-DAQ-64-3M/KPXI-DAQ-96-3M)**

**NOTE** The last 4 digital codes are *SDI<3..0>* and are supported in KPXI-DAQ-64-3M only

Midscale +1LSB	5.00244V	2.50122V	1.25061V	001X
Midscale	5V	2.5V	1.25V	000X
Midscale -1LSB	4.9976V	2.4988V	1.2494V	FFF
-FSR	0V	0V	0V	800X

**KPXI-DAQ-64-500K/KPXI-DAQ-64-250K AI Data Format**

The data format of the acquired 16-bit A/D data is 2's Complement coding. Table 3-5 and Table 3-6 shows the valid input ranges and the ideal transfer characteristics.

Table 3-5  
**Bipolar analog input range (KPXI-DAQ-64-500K/KPXI-DAQ-64-250K)**

Description	Bipolar Analog Input Range				Digital code
	±10V	±5V	±2.5V	±1.25V	
Full-scale Range	±10V	±5V	±2.5V	±1.25V	
Least significant bit	305.2uV	152.6uV	76.3uV	38.15uV	
FSR-1LSB	9.999695V	4.999847V	2.499924V	1.249962V	7FFF
Midscale +1LSB	305.2uV	152.6uV	76.3uV	38.15uV	0001
Midscale	0V	0V	0V	0V	0000
Midscale -1LSB	-305.2uV	-152.6uV	-76.3uV	-38.15uV	FFFF
-FSR	-10V	-5V	-2.5V	-1.25V	8000

Table 3-6  
**Unipolar analog input range (KPXI-DAQ-64-500K/KPXI-DAQ-64-250K)**

Description	Unipolar Analog Input Range				Digital code
	0V to 10V	0 to +5V	0 to +2.5V	0 to +1.25V	
Full-scale Range	0V to 10V	0 to +5V	0 to +2.5V	0 to +1.25V	
Least significant bit	152.6uV	76.3uV	38.15uV	19.07uV	
FSR-1LSB	9.999847V	4.999924V	2.499962V	1.249981V	7FFF
Midscale +1LSB	5.000153V	2.500076V	1.250038V	0.625019V	0001
Midscale	5V	2.5V	1.25V	0.625V	0000
Midscale -1LSB	4.999847V	2.499924V	1.249962V	0.624981V	FFFF
-FSR	0V	0V	0V	0V	8000

**Software conversion with polling data transfer acquisition mode (Software Polling)**

This is the easiest way to acquire a single A/D data. The A/D converter starts one conversion whenever the dedicated software command is executed. Then the software would poll the conversion status and read the A/D data back when it is available.

This method is very suitable for applications that needs to process A/D data in real time. Under this mode, the timing of the A/D conversion is fully controlled under software. However, it is difficult to control the A/D conversion rate.

**Specifying Channels, Gains, and input configurations in the Channel Gain Queue**

In Software Polling and Programmable Scan Acquisition mode, the channel, gain, polarity, and input configuration (RSE, NRSE, or DIFF) can be specified in the **Channel Gain Queue**. You can fill the channel number in the Channel Gain Queue in any order. The channel order of acquisition

will be the same as the order you set in the Channel Gain Queue. Therefore, you can acquire data with user-defined channel orders and with different settings on each channel.

When the specified channels have been sampled from the first data to the last data in the Channel Gain Queue, the settings in Channel Gain Queue are maintained. You don't need to re-configure the Channel Gain Queue if you want to keep on sampling data in the same order. The maximum number of entries you can set in the Channel Gain Queue is 512.

**Example:**

First you can set entries in Channel Gain Queue:

Ch3 with bipolar  $\pm 10V$ , RSE connection

Ch1 with bipolar  $\pm 2.5V$ , DIFF connection

Ch2 with unipolar 5V, NRSE connection

Ch1 with bipolar  $\pm 2.5V$ , DIFF connection

If you read 10 data by software polling method

Then the acquisition sequence of channels is: 3, 1, 2, 1, 3, 1, 2, 1, 3, 1

**Programmable scan acquisition mode****Scan Timing and Procedure**

It's recommended that this mode be used if your applications need a fixed and precise A/D sampling rate. You can accurately program the period between conversions of individual channels. There are at least 4 counters, which need to be specified:

SI\_counter (24 bit): Specify the **Scan Interval** =  $SI\_counter / \text{Timebase}$

SI2\_counter (16 bit): Specify the data **Sampling Interval** =  $SI2\_counter / \text{Timebase}$

PSC\_counter (24 bit): Specify **Post Scan Counts** after a trigger event

NumChan\_counter (9 bit): Specify the Number of samples per scan

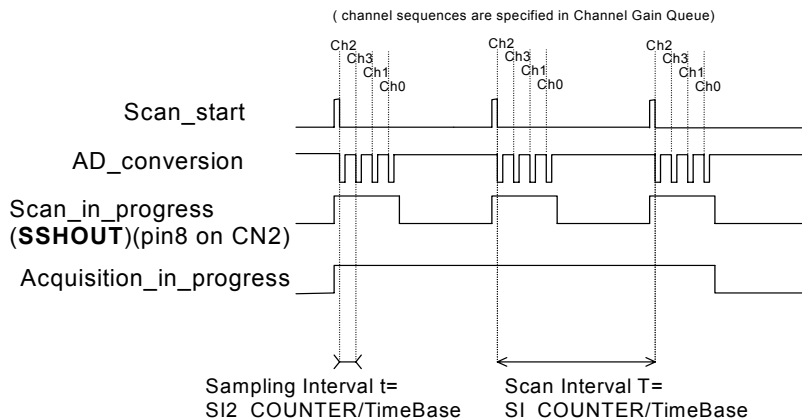
The acquisition timing and the meaning of the 4 counters are illustrated in [Figure 3-11](#).

**Timebase Clock Source**

In scan acquisition mode, all the A/D conversions start on the output of counters, which use **Timebase** as the clock source. By software you can specify the Timebase to be either an internal clock source (on-board 40MHz) or an external clock input (EXTTIMEBASE) on CN2. The external clock is useful when you want to acquire data at rates not available with the internal A/D sample clock. The external clock source should generate TTL-compatible continuous clocks, and the maximum frequency is 40MHz while the minimum is 1MHz.

Figure 3-11  
Scan timing

3 Scans, 4 Samples per scan  
(PSC\_Counter=3, NumChan\_Counter=4)



There are 4 trigger modes to start the scan acquisition, please refer to [Trigger Modes](#) for the details. The data transfer mode will be discussed in [Bus-mastering DMA Data Transfer](#).

- NOTE**
1. The maximum A/D sampling rate is 3MHz for KPXI-DAQ-64-3M/KPXI-DAQ-96-3M, 500kHz for KPXI-DAQ-64-500K and 250kHz for KPXI-DAQ-64-250K. Therefore, the minimum setting for SI2\_counter is 14 for KPXI-DAQ-64-3M/KPXI-DAQ-96-3M, 80 for KPXI-DAQ-64-500K and 160 for KPXI-DAQ-64-250K while using the internal Timebase.
  2. The SI\_counter is a 24-bit counter and the SI2\_counter is a 16-bit counter. Therefore, the maximum scan interval using the internal Timebase =  $2^{24}/40M \text{ s} = 0.419\text{s}$ , and the maximum sampling interval between 2 channels using the internal Timebase =  $2^{16}/40M \text{ s} = 1.638\text{ms}$ .
  3. The scan interval can't be smaller than the product of the data sampling interval and the NumChan\_counter value. The relationship can be represented as:  
 $SI\_counter \geq SI2\_counter * NumChan\_counter$ .

**Scan with SSH (KPXI-DAQ-96-3M doesn't support this function)**

You can send the SSHOUT signal on CN2 to an external S&H circuits to sample and hold all signals if you want to simultaneously sample all channels in a scan, as illustrated in [Figure 3-11](#).

- NOTE** The SSHOUT signal is sent to external S&H circuits to hold the analog signal. Users must implement external S&H circuits on their own to carry out the S&H function. There are no on-board S&H circuits.

**Specifying Channels, Gains, and input configurations in the Channel Gain Queue**

Like software polling acquisition mode, the channel, gain, and input configurations can be specified in the **Channel Gain Queue** under the scan acquisition mode. Please refer to [Specifying Channels, Gains, and input configurations in the Channel Gain Queue](#). Note that in scan acquisition mode the number of entries in the Channel Gain Queue is normally equivalent to the value of NumChan\_counter (that is, the number of samples per scan).

**Example:**

Set:

SI2\_counter = 160

SI\_counter = 640

PSC\_counter = 3

NumChan\_counter = 4

Timebase = Internal clock source

Channel entries in the Channel Gain Queue: ch1, ch2, ch0, ch2

Then:

Acquisition sequence of channels: 1, 2, 0, 2, 1, 2, 0, 2, 1, 2, 0, 2

Sampling Interval =  $160/40\text{M s} = 4\text{ us}$ Scan Interval =  $640/40\text{M s} = 16\text{ us}$ 

Equivalent sampling rate of ch0, ch1: 62.5kHz

Equivalent sampling rate of ch2: 125kHz

**Trigger Modes**

KPXI-DAQ module provides 3 trigger sources (internal software, external analog and digital trigger sources). You must select one of them as the source of the trigger event. A trigger event occurs when the specified condition is detected on the selected trigger source (For example, a rising edge on the external digital trigger input).

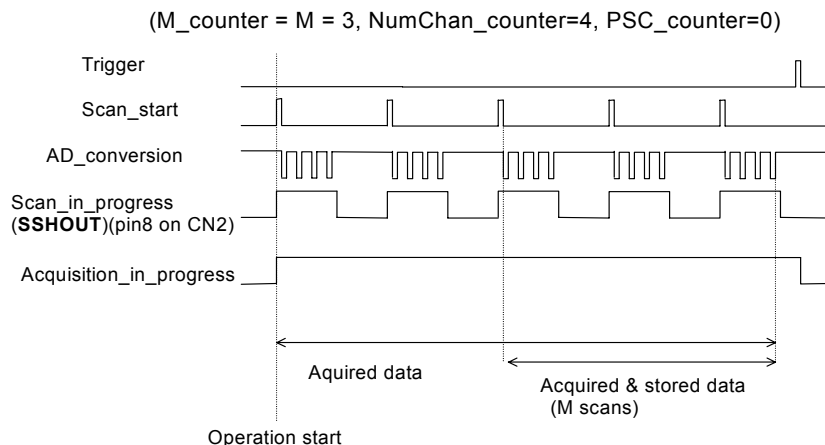
There are 4 trigger modes (pre-trigger, post-trigger, middle-trigger, and delay-trigger) working with the 4 trigger sources to initiate different scan data acquisition timing when a trigger event occurs. They are described as follows. For information of trigger sources, please refer to [Trigger Sources](#).

**Pre-Trigger Acquisition**

Use pre-trigger acquisition in applications where you want to collect data before a trigger event. The A/D starts when you execute the specified function calls to begin the operation, and it stops when the external trigger event occurs. Users must program the value M in **M\_counter** (16bit) to specify the amount of stored scanned data before the trigger event. If an external trigger occurs after M scans of data are converted, the program only stores the last M scans of data, as illustrated in [Figure 3-12](#), where M\_counter = M = 3, NumChan\_counter = 4, PSC\_counter = 0. The total stored amount of data = NumChan\_counter \* M\_counter = 12.

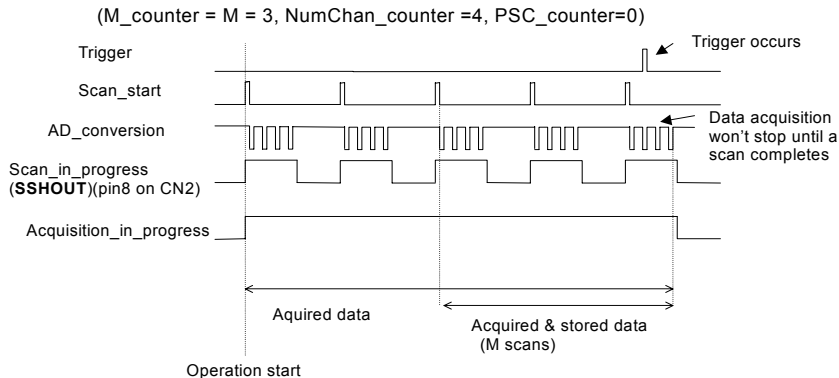


Figure 3-12  
**Pre-trigger (trigger occurs after M scans)**



Note that if a trigger event occurs when a scan is in progress, the data acquisition won't stop until the scan completes, and the stored M scans of data includes the last scan. **Therefore, the first stored data will always be the first channel entry of a scan (that is, the first channel entry in the Channel Gain Queue if the number of entries in the Channel Gain Queue is equivalent to the value of NumChan\_counter), no matter when a trigger signal occurs,** as illustrated in [Figure 3-13](#), where M\_counter = M = 3, NumChan\_counter = 4, PSC\_counter = 0.

Figure 3-13  
**Pre-trigger (trigger with scan is in progress)**



When a trigger signal occurs before the first M scans of data are converted, the amount of stored data could be fewer than the originally specified amount in NumChan\_counter \* M\_counter, as illustrated in [Figure 3-14](#). This situation can be avoided by setting **M\_enable**. If **M\_enable** is set to 1, the trigger signal will be ignored until the first M scans of data are converted, and it assures user can get M scans of data under pre-trigger mode, as illustrated in [Figure 3-15](#). However, if **M\_enable** is set to 0, the trigger signal will be accepted in any time, as illustrated in [Figure 3-14](#). Note that the total amount of stored data is still always a multiple of NumChan\_counter (number of samples per scan) because the data acquisition won't stop until a scan is completed.

Figure 3-14  
**Pre-trigger with  $M\_enable = 0$  (trigger occurs before  $M$  scans)**

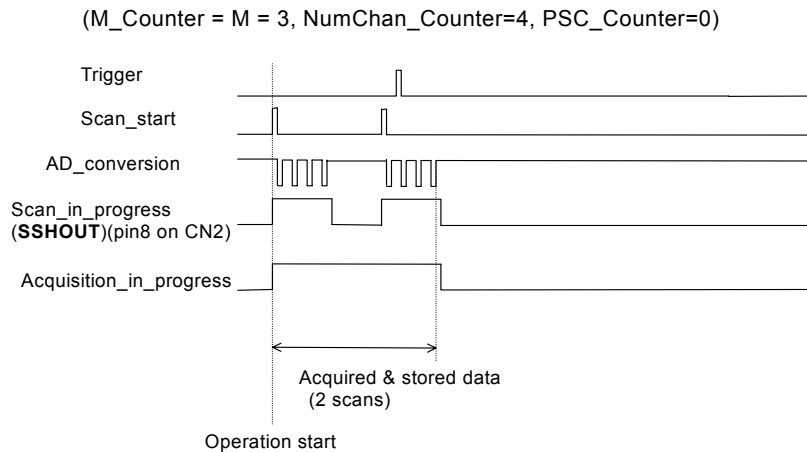
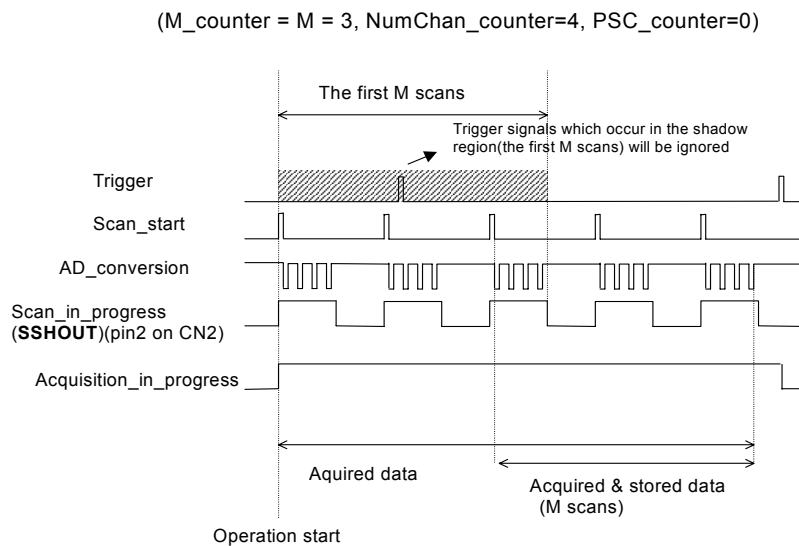


Figure 3-15  
**Pre-trigger with  $M\_enable = 1$**



**NOTE** The  $PSC\_counter$  is set to 0 in pre-trigger acquisition mode.

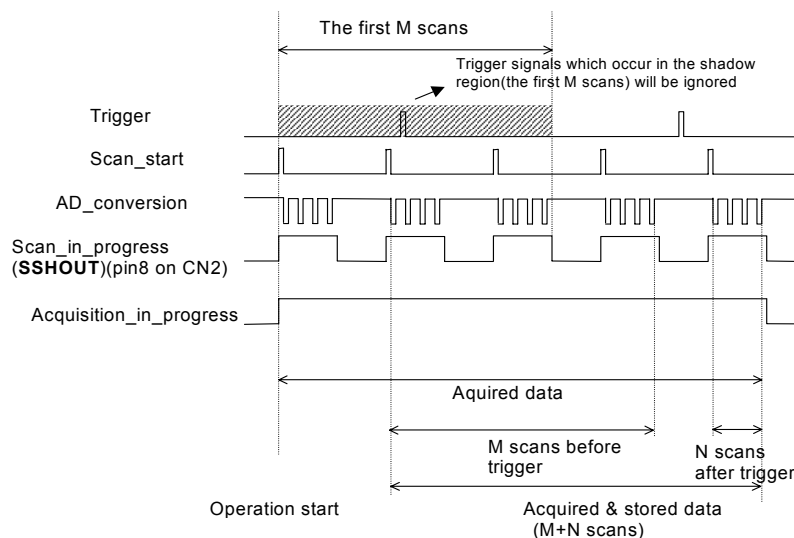
### Middle-Trigger Acquisition

Use middle-trigger acquisition in applications where you want to collect data before and after a trigger event. The number of scans ( $M$ ) stored before the trigger is specified by  $M\_counter$ , while the number of scans ( $N$ ) after the trigger is specified by  $PSC\_counter$ .

Like pre-trigger mode, the number of stored data could be fewer than the specified amount of data ( $NumChan\_counter * (M+N)$ ) if an external trigger occurs before  $M$  scans of data are converted. The  $M\_enable$  bit in middle-trigger mode takes the same effect as in pre-trigger mode. If  $M\_enable$  is set to 1, the trigger signal will be ignored until the first  $M$  scans of data are converted, and it assures users can get  $(M+N)$  scans of data under middle-trigger mode. However, if  $M\_enable$  is set to 0, the trigger signal will be accepted at any time. Figure 3-16 shows the acquisition timing with  $M\_enable=1$ .

Figure 3-16  
**Middle trigger with M\_enable = 1**

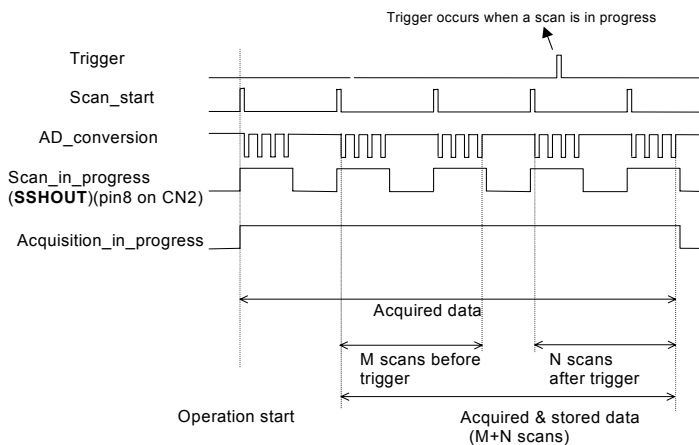
(M\_Counter=M=3, NumChan\_Counter=4, PSC\_Counter=N=1)



If a trigger event occurs when a scan is in progress, the stored N scans of data would include this scan. **And the first stored data will always be the first channel entry of a scan**, as illustrated in Figure 3-17.

Figure 3-17  
**Middle trigger (trigger occurs when a scan is in progress)**

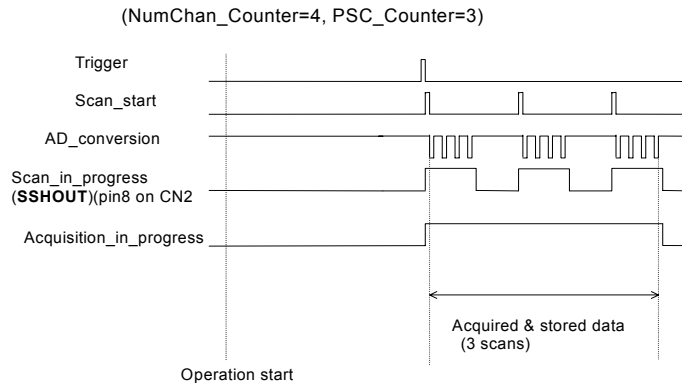
(M\_Counter=M=2, NumChan\_Counter=4, PSC\_Counter=N=2)



**Post-Trigger Acquisition**

Use post-trigger acquisition in applications where you want to collect data after a trigger event. The number of scans after the trigger is specified in PSC\_counter, as illustrated in Figure 3-18. The total acquired data length = NumChan\_counter \* PSC\_counter.

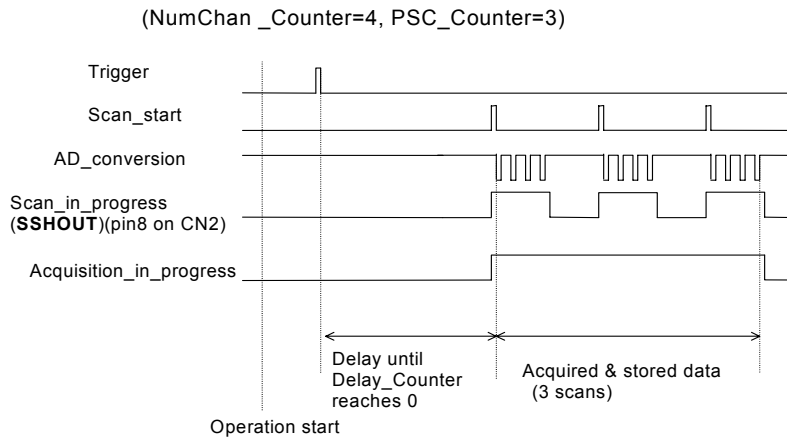
Figure 3-18  
Post trigger



**Delay Trigger Acquisition**

Use delay trigger acquisition in applications where you want to delay the data collecting process after the occurrence of a specified trigger event. The delay time is controlled by the value, which is pre-loaded in the **Delay\_counter** (16bit). The counter counts down on the rising edge of the Delay\_counter clock source after the trigger condition is met. The clock source is software programmed and can be either the Timebase clock (40MHz) or the A/D sampling clock (Timebase/SI2\_counter). When the count reaches 0, the counter stops and the board starts to acquire data. The total acquired data length = NumChan\_counter \* PSC\_counter.

Figure 3-19  
Delay trigger



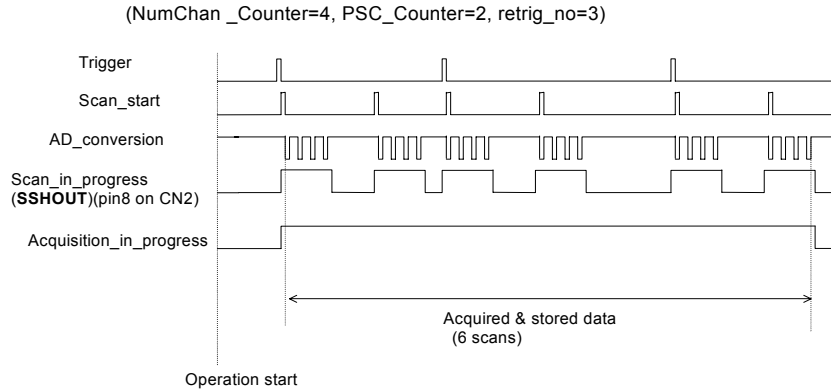
**NOTE** When the Delay\_counter clock source is set to Timebase, the maximum delay time =  $2^{16} / 40M$  s = 1.638ms, and when set to A/D sampling clock, the maximum delay time can be higher ( $2^{16} * SI2\_counter / 40M$ ).

**Post-Trigger or Delay-trigger Acquisition with retrigger**

Use post-trigger or delay-trigger acquisition with re-trigger function in applications where you want to collect data after several trigger events. The number of scans after each trigger is specified in PSC\_counter, and users could program **Retrig\_no** to specify the re-trigger numbers. Figure 3-20 illustrates an example. In this example, 2 scans of data is acquired after the first trigger signal, then the board waits for the re-trigger signal (re-trigger signals which occur before the first 2 scans of data acquired will be ignored). When the re-trigger signal occurs, the board scans 2 more scans

of data. The process repeats until the specified amount of re-trigger signals are detected. The total acquired data length = NumChan\_counter \* PSC\_counter \* Retrig\_no.

Figure 3-20  
**Post trigger with retrigger**



**Bus-mastering DMA Data Transfer**

PCI bus-mastering DMA is necessary for high speed DAQ in order to utilize the maximum PCI bandwidth. The bus-mastering controller, which is built in the PLX IOP-480 PCI controller, controls the PCI bus when it becomes the master of the bus. Bus mastering reduces the size of the on-board memory and reduces the CPU loading because data is directly transferred to the computer’s memory without host CPU intervention.

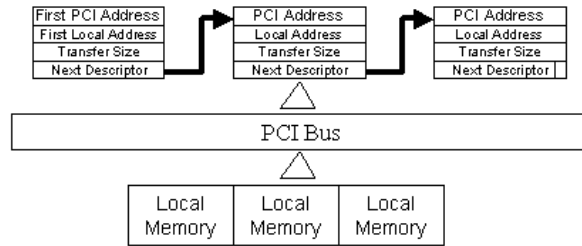
Bus-mastering DMA provides the fastest data transfer rate on PCI-bus. Once the analog input operation starts, control returns to your program. The hardware temporarily stores the acquired data in the on-board AD Data FIFO and then transfers the data to a user-defined DMA buffer memory in the computer. Please note that even when the acquired data length is less than the Data FIFO, the AD data will not be kept in the Data FIFO but directly transferred into host memory by the bus-mastering DMA.

By using a high-level programming library for high speed DMA data acquisition, users simply need to assign the sampling period and the number of conversion into their specified counters. After the AD trigger condition is matched, the data will be transferred to the system memory by the bus-mastering DMA.

The PCI controller also supports the function of scatter/gather bus mastering DMA, which helps the users to transfer large amounts of data by linking all the memory blocks into a continuous linked list.

In a multi-user or multi-tasking OS, like Microsoft Windows, it is difficult to allocate a large continuous memory block to do the DMA transfer. Therefore, the PLX IOP-480 provides the function of scatter /gather or chaining mode DMA to link the non-continuous memory blocks into a linked list so that users can transfer very large amounts of data without being limited by the fragmentation of memory. Users can configure the linked list for the input DMA channel or the output DMA channel. Figure 3-21 shows a linked list that is constructed by three DMA descriptors. Each descriptor contains a PCI address, a local address, a transfer size, and the pointer to the next descriptor. Users can allocate many small size memory blocks and chain their associative DMA descriptors altogether by their application programs. KPXI-DAQ module software driver provides simple settings of the scatter/gather function, and some sample programs are also provided on the CD.

Figure 3-21  
**Scatter/gather DMA for data transfer**



In non-chaining mode, the maximum DMA data transfer size is 2M double words (8M bytes). However, by using chaining mode, scatter/gather, there is no limitation on DMA data transfer size. Users can also link the descriptor nodes circularly to achieve a multi-buffered mode DMA.

### D/A Conversion

**NOTE** *KPXI-DAQ-96-3M does not support this function.*

There are 2 channels of 12-bit D/A output available in the KPXI-DAQ module. When using D/A converters, users should assign and control the D/A converter reference sources for the D/A operation mode and D/A channels. Users could also select the output polarity: unipolar or bipolar.

The reference selection control lets users fully utilize the multiplying characteristics of the D/A converters. Internal 10V reference and external reference inputs are available in the KPXI-DAQ module. The range of the D/A output is directly related to the reference. The digital codes that are updated to the D/A converters will multiply with the reference to generate the analog output. While using internal 10V reference, the full range would be  $-10V \sim +9.9951V$  in the bipolar output mode, and  $0V \sim 9.9976V$  in the unipolar output mode. While using an external reference, users can reach different output ranges by connecting different references. For example, if connecting a DC  $-5V$  with the external reference, then the users can get a full range from  $-4.9976V$  to  $+5V$  in the bipolar output with inverting characteristics due to the negative reference voltage. Users could also have an amplitude modulated (AM) output by feeding a sinusoidal signal into the reference input. The range of the external reference should be within  $\pm 10V$ . [Table 3-7](#) and [Table 3-8](#) illustrates the relationship between digital code and output voltages.

Table 3-7

**Bipolar output code table(Vref=10V if internal reference is selected)**

Digital Code	Analog Output
111111111111	$V_{ref} * (2047/2048)$
100000000001	$V_{ref} * (1/2048)$
100000000000	0V
011111111111	$-V_{ref} * (1/2048)$
000000000000	$-V_{ref}$

Table 3-8  
**Unipolar output code table (Vref=10V if internal reference is selected)**

Digital Code	Analog Output
111111111111	Vref * (4095/4096)
100000000000	Vref * (2048/4096)
000000000001	Vref * (1/4096)
000000000000	0V

The D/A conversion is initiated by a trigger source. Users must decide how to trigger the D/A conversion. The data output will start when a trigger condition is met. Before the start of D/A conversion, D/A data is transferred from PC’s main memory to a buffering Data FIFO.

There are two modes of the D/A conversion: Software Update and Timed Waveform Generation are described, including timing, trigger source control, trigger modes and data transfer methods. **Either mode may be applied to D/A channels independently.** You can software update DA CH0 while generate timed waveforms on CH1 at the same time.

**Software Update**

This is the easiest way to generate D/A output. First, users should specify the D/A output channels, set output polarity: unipolar or bipolar, and reference source: internal 10V or external AOEXTREF. Then update the digital values into D/A data registers through a software output command.

**Timed Waveform Generation**

This mode can provide your applications with a precise D/A output with a fixed update rate. It can be used to generate an infinite or finite waveform. You can accurately program the update period of the D/A converters.

The D/A output timing is provided through a combination of counters in the FPGA on board. There are in total 5 counters to be specified. These counters are:

UI\_counter(24 bits): specify the DA **Update Interval** = CHUI\_counter/Timebase.

UC\_counter(24 bits): specify the total **Update Counts** in a single waveform

IC\_counter(24 bits): specify the **Iteration Counts** of waveform.

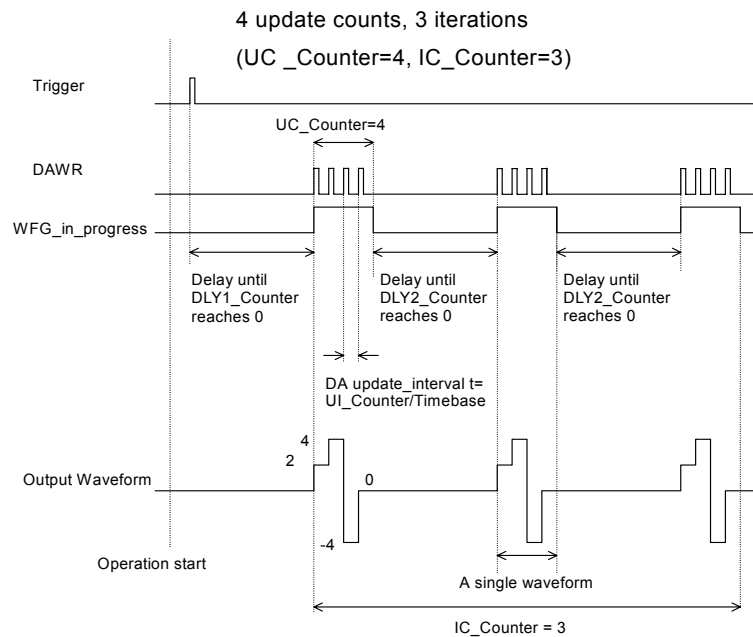
DLY1\_counter(16 bits): specify the **Delay** from the trigger to the first update start.

DLY2\_counter(16 bits): specify the **Delay** between two consecutive waveform generations.

Figure 3-22 shows the typical D/A timing diagram. D/A updates its output on each rising edge of DAWR. The meaning of the counters above is discussed more in the following sections. For more information of Timebase, please refer to [Timebase Clock Source](#).

Figure 3-22  
**Typical D/A timing of waveform generation**

**NOTE** *Figure 3-22 assumes the data in the buffer is as follows: 2V, 4V, -4V, and 0V.*



**NOTE** *The maximum D/A update rate is 1MHz. Therefore; the minimum setting of  $UI\_counter$  is 40 while using the internal Timebase (40MHz).*

## Trigger Modes

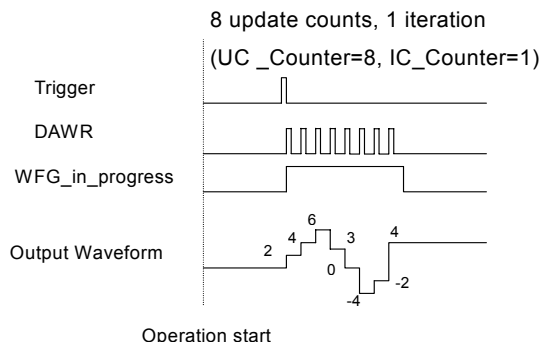
### Post-Trigger Generation

Use post trigger when you want to perform DA waveform right after a trigger event occurs. In this trigger mode  $DLY1\_Counter$  is not used and you don't need to specify it. Figure 3-23 shows a single waveform generated right after a trigger signal is detected. The trigger signal could come from a software command, an analog trigger or a digital trigger. Please refer to [Trigger Sources](#) for detailed information.



Figure 3-23  
**Post trigger waveform generation**

**NOTE** *Figure 3-23 assumes the data in the buffer is as follows: 2V, 4V, 6V, 3V, 0V, -4V, -2V, and 4V.*

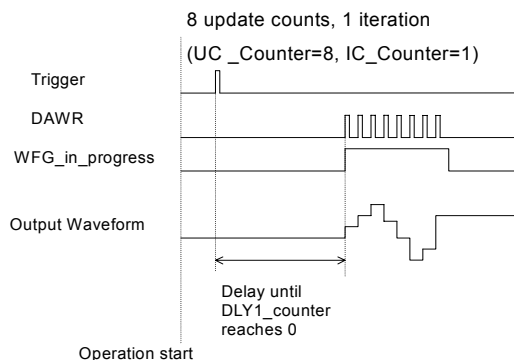


**Delay-Trigger Generation**

Use delay trigger when you want to delay the waveform generation after a trigger event. In [Figure 3-24](#), DA\_DLY1\_counter determines the delay time from the trigger signal to the start of the waveform generation. DLY1\_counter counts down on the rising edge of its clock source after the trigger condition is met. When the count reaches 0, the counter stops and the KPXI-DAQ module starts the waveform generation. This DLY1\_Counter is 16-bit's wide and users can set the delay time in units of TIMEBASE (delay time = DLY1\_Counter/TIMEBASE) or in units of update period (delay time DLY1\_Counter \* UI\_counter/TIMEBASE), such that the delay time can reach a wider range.

Figure 3-24  
**Delay trigger waveform generation**

**NOTE** *Figure 3-24 assumes the data in data buffer is as follows: 2V, 4V, 6V, 3V, 0V, -4V, -2V, and 4V.*

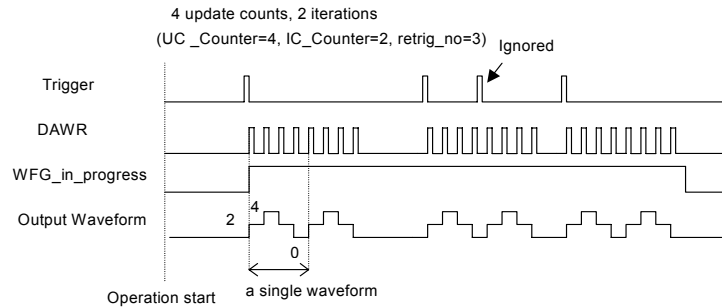


**Post-Trigger or Delay-Trigger with Re-trigger**

Use post-trigger or delay-trigger with re-trigger function when you want to generate waveform after more than one trigger events. The re-trigger function can be enabled or disabled by software setting. In [Figure 3-25](#), each trigger signal will generate 2 single waveforms (since IC\_Counter = 2), and you can set **Retrig\_no** to specify the number of the accepted re-trigger signals. Note that a trigger would be ignored if it occurs during waveform generation.

Figure 3-25  
**Re-triggered waveform generation with Post-trigger and DLY2\_Counter = 0**

**NOTE** Figure 3-25 assumes the data in the buffer is as follows: 2V, 4V, 2V, and 0V.



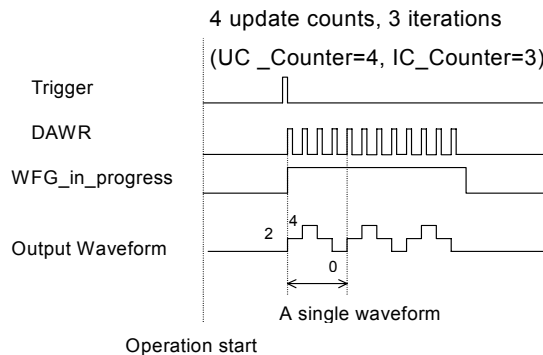
**Iterative Waveform Generation**

Set IC\_Counter in order to generate iterative waveforms from the data of a single waveform. The counter stores the iteration number, and the iterations could be finite (Figure 3-26) or infinite (Figure 3-27). Note that in infinite mode the waveform generation won't stop until software stop function is executed, and IC\_Counter is still meaningful when stop mode III is selected. Please refer to Stop Modes of Scan Update for details.

A data FIFO on board is used to buffer the digital data for DA output. If the data size of a single waveform specified (That is, Update Counts in UC\_Counter) is less than the FIFO size, after initially transferring the data from host PC memory to the FIFO on board, the data in FIFO will be automatically re-transmitted whenever a single waveform is completed. Therefore, it won't occupy the PCI bandwidth when the iterative waveforms are performed. However, if the data size of a single waveform specified is more than the FIFO size, it needs to intermittently perform DMA to transfer data from host PC memory to the FIFO on board when the iterative waveforms are performed and occupies PCI bandwidth. The data FIFO size on the KPXI-DAQ module is 1024 (words) when one DA channel is enabled, and 512 (words) when both DA channels are enabled.

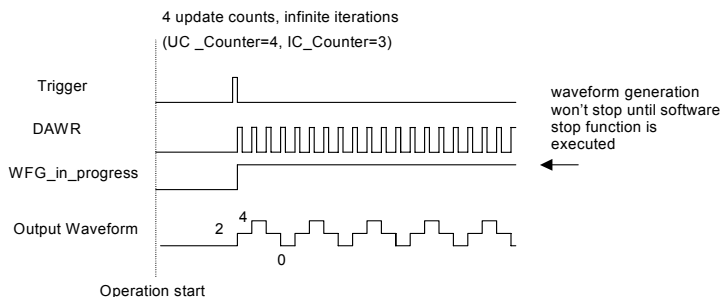
Figure 3-26  
**Finite iterative waveform generation with Post-trigger and DLY2\_Counter = 0**

**NOTE** Figure 3-26 assumes the data in the buffer is as follows: 2V, 4V, 2V, and 0V.



**Figure 3-27**  
**Infinite iterative waveform generation with Post-trigger and DLY2\_Counter = 0**

**NOTE** *Figure 3-27 assumes the data in the buffer is as follows: 2V, 4V, 2V, and 0V.*



**Delay2 in iterative Waveform Generation**

To stretch out the flexibility of the D/A waveform generation, we add a DLY2 Counter to separate 2 consecutive waveforms in iterative waveform generation. The time between two waveforms is assigned by setting the value of DLY2\_Counter. The DLY2\_Counter counts down after a complete waveform generation, and when it counts down to zero, the next waveform generation will start. This DLY2\_Counter is 16-bits wide and users can set the delay time in the unit of Timebase (delay time = DLY2\_Counter/Timebase) or in the unit of update period (delay time = DLY2\_Counter \* UI\_Counter/Timebase), such that the delay time could reach a wide range.

**Stop Modes of Scan Update**

You can call software stop function to stop waveform generation while it is still in progress. Three stop modes are provided for timed waveform generation. You can apply these 3 modes to stop waveform generation regardless of whether infinite or finite waveform generation mode is selected.

Figure 3-28 illustrates an example for stop mode I, in this mode the waveform stops immediately when software command is asserted.

In stop mode II, after a software stop command is given, the waveform generation won't stop until a complete single waveform is finished. See Figure 3-29 for an example, since UC\_Counter is set to 4, the total DA updates counts (that is, number of pulses of DAWR signal) must be a multiple of 4 (update counts = 20 in this example).

In stop mode III, after a software stop command is given, the waveform generation won't stop until the performed number of waveforms is a multiple of IC\_Counter. See Figure 3-30 for an example, since IC\_Counter is set to 3, the total generated waveforms must be a multiple of 3 (waveforms = 6 in this example), and the total DA update counts must be a multiple of 12 (UC\_Counter \* IC\_Counter). You can compare these three figures for their differences.

Figure 3-28  
**Stop mode I**

**NOTE** Figure 3-28 assumes the data in the buffer is as follows: 2V, 4V, 2V, and 0V.

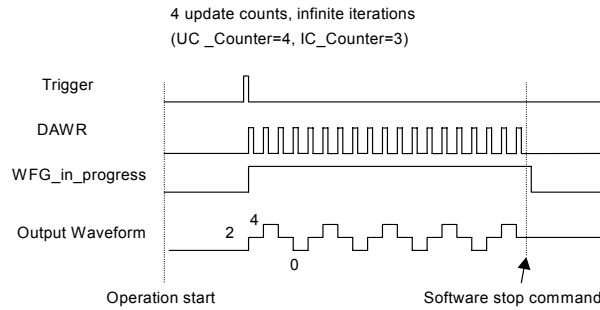


Figure 3-29  
**Stop mode II**

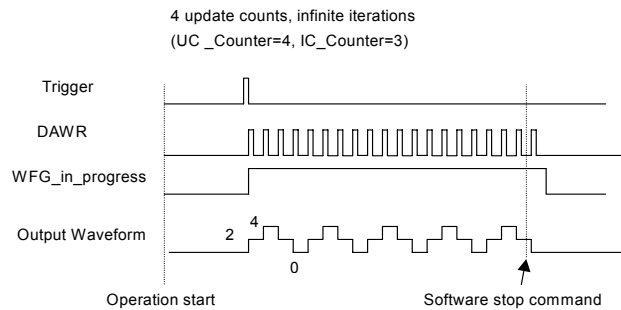
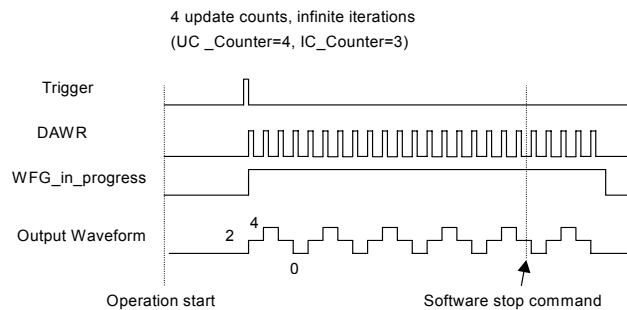


Figure 3-30  
**Stop mode III**



## Digital I/O

The KPXI-DAQ module contains 24-lines of general-purpose digital I/O (GPIO), which is provided through a 82C55A chip.

The 24-lines GPIO are separated into three ports: Port A, Port B and Port C. Port A, Port B, Port C high nibble (bit-4 to bit-7), and low nibble (bit 0 to bit 3) can be programmed to be input or output individually. At system startup and reset, all the I/O pins are all reset to be input configuration, that is, high impedance.

KPXI-DAQ-64-3M also provides 4 digital inputs (SDI from CN2), which are sampled simultaneously with the analog signal input and stored with the 12-bit AD data. Please refer to [Synchronous Digital Inputs \(for KPXI-DAQ-64-3M only\)](#) for the details.

## General Purpose Timer/Counter Operation

**NOTE** *KPXI-DAQ-96-3M doesn't support this function.*

Two independent 16-bit up/down timer/counter are designed within FPGA for various applications. They have the following features:

- Count up/down controlled by hardware or software
- Programmable counter clock source (internal or external clock up to 10MHz)
- Programmable gate selection (hardware or software control)
- Programmable input and output signal polarities (high active or low active)
- Initial Count can be loaded from software
- Current count value can be read-back by software without affecting circuit operation

### Timer/Counter functions basics

Each timer/counter has three inputs that can be controlled via hardware or software. They are clock input (GPTC\_CLK), gate input (GPTC\_GATE), and up/down control input (GPTC\_UPDOWN). The GPTC\_CLK input provides a clock source input to the timer/counter. Active edges on the GPTC\_CLK input make the counter increment or decrement. The GPTC\_UPDOWN input controls whether the counter counts up or down. The GPTC\_GATE input is a control signal which acts as a counter enable or a counter trigger signal under different applications.

The output of timer/counter is GPTC\_OUT. After power-up, GPTC\_OUT is pulled high by a pulled-up resistor about 10K ohms. Then GPTC\_OUT goes low after the KPXI-DAQ module initialization.

All the polarities of input/output signals can be programmed by software. In this section, for easy explanation, all GPTC\_CLK, GPTC\_GATE, and GPTC\_OUT are assumed to be active high or rising-edge triggered in the figures.

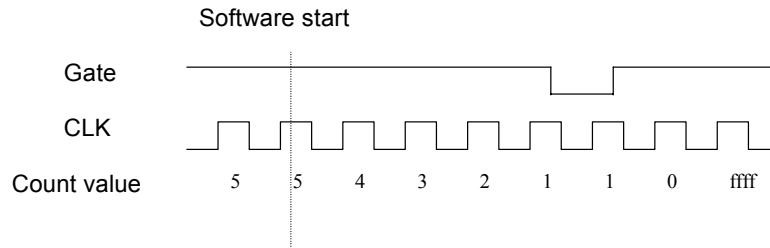
### General Purpose Timer/Counter modes

Eight programmable timer/counter modes are provided. All modes start operating following a software-start signal that is set by the software. The GPTC software reset initializes the status of the counter and re-loads the initial value to the counter. The operation remains halted until the software-start is re-executed. The operating theories under different modes are described as below.

#### Mode1: Simple Gated-Event Counting

In this mode, the counter counts the number of pulses on the GPTC\_CLK after the software-start. Initial count can be loaded from software. Current count value can be read-back by software any time without affecting the counting. GPTC\_GATE is used to enable/disable counting. When GPTC\_GATE is inactive, the counter halts the current count value. [Figure 3-31](#) illustrates the operation with initial count = 5, count-down mode.

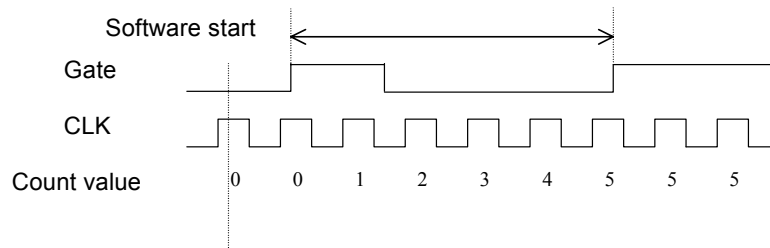
Figure 3-31  
**Mode 1 operation**



**Mode2: Single Period Measurement**

In this mode, the counter counts the period of the signal on GPTC\_GATE in terms of GPTC\_CLK. Initial count can be loaded from software. After the software-start, the counter counts the number of active edges on GPTC\_CLK between two active edges of GPTC\_GATE. After the completion of the period interval on GPTC\_GATE, GPTC\_OUT outputs high and then current count value can be read-back by software. Figure 3-32 illustrates the operation where initial count = 0, count-up mode.

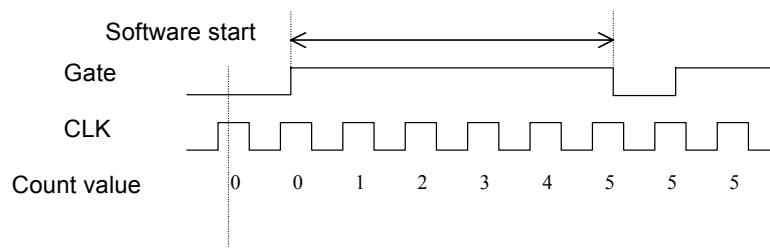
Figure 3-32  
**Mode 2 operation**



**Mode 3: Single Pulse-width Measurement**

In this mode the counter counts the pulse-width of the signal on GPTC\_GATE in terms of GPTC\_CLK. Initial count can be loaded from software. After the software-start, the counter counts the number of active edges on GPTC\_CLK when GPTC\_GATE is in its active state. After the completion of the pulse-width interval on GPTC\_GATE, GPTC\_OUT outputs high and then current count value can be read-back by software. Figure 3-33 illustrates the operation where initial count = 0, count-up mode.

Figure 3-33  
**Mode 3 operation**

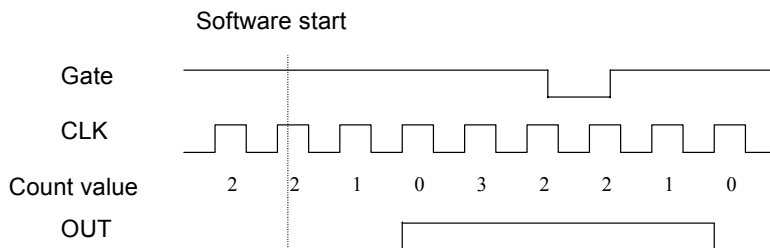


**Mode 4: Single Gated Pulse Generation**

This mode generates a single pulse with programmable delay and programmable pulse-width following the software-start. The two programmable parameters could be specified in terms of

periods of the GPTC\_CLK input by software. GPTC\_GATE is used to enable/disable counting. When GPTC\_GATE is inactive, the counter halts the current count value. Figure 3-34 illustrates the generation of a single pulse with a pulse delay of two and a pulse-width of four.

Figure 3-34  
**Mode 4 operation**



**Mode5: Single Triggered Pulse Generation**

This function generates a single pulse with programmable delay and programmable pulse-width following an active GPTC\_GATE edge. You could specify these programmable parameters in terms of periods of the GPTC\_CLK input. Once the first GPTC\_GATE edge triggers the single pulse, GPTC\_GATE takes no effect until the software-start is re-executed. Figure 3-35 illustrates the generation of a single pulse with a pulse delay of two and a pulse-width of four.

Figure 3-35  
**Mode 5 operation**

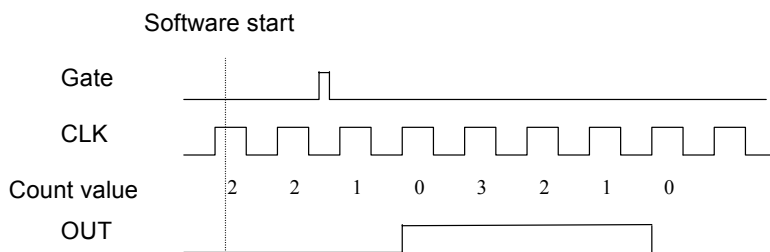
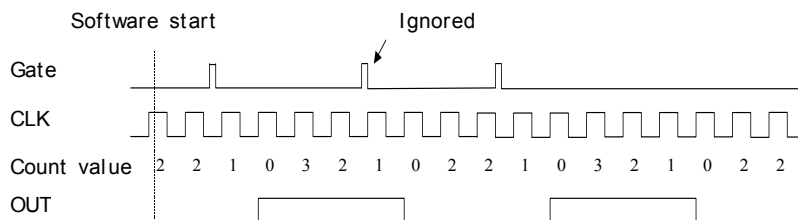


Figure 38:

**Mode6: Re-triggered Single Pulse Generation**

This mode is similar to mode5 except that the counter generates a pulse following every active edge of GPTC\_GATE. After the software-start, every active GPTC\_GATE edge triggers a single pulse with programmable delay and pulse-width. Any GPTC\_GATE triggers that occur when the prior pulse is not completed would be ignored. Figure 3-36 illustrates the generation of two pulses with a pulse delay of two and a pulse-width of four.

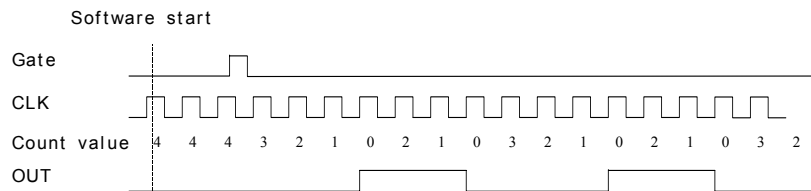
Figure 3-36  
**Mode 6 operation**



### Mode7: Single Triggered Continuous Pulse Generation

This mode is similar to mode5 except that the counter generates continuous periodic pulses with programmable pulse interval and pulse-width following the first active edge of GPTC\_GATE. Once the first GPTC\_GATE edge triggers the counter, GPTC\_GATE takes no effect until the software-start is re-executed. [Figure 3-37](#) illustrates the generation of two pulses with a pulse delay of four and a pulse-width of three.

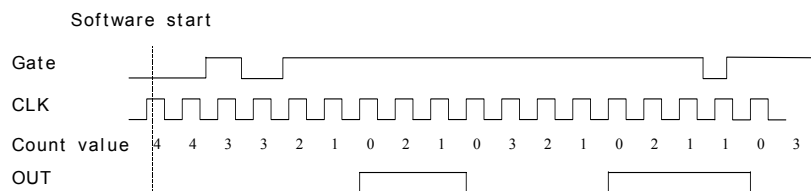
Figure 3-37  
Mode 7 operation



### Mode8: Continuous Gated Pulse Generation

This mode generates periodic pulses with programmable pulse interval and pulse-width following the software-start. GPTC\_GATE is used to enable/disable counting. When GPTC\_GATE is inactive, the counter halts the current count value. [Figure 3-38](#) illustrates the generation of two pulses with a pulse delay of four and a pulse-width of three.

Figure 3-38  
Mode 8 operation



## Trigger Sources

The KPXI-DAQ series provides flexible trigger selections. In addition to the internal software trigger, KPXI-DAQ module also supports external analog, digital triggers and SSI triggers. Users can configure the trigger source by software for A/D and D/A processes individually. **Note that the A/D and the D/A conversion share the same analog trigger.**

### Software-Trigger

This trigger mode does not need any external trigger source. The trigger asserts right after you execute the specified function calls to begin the operation. A/D and D/A processes can receive an individual software trigger.

### External Analog Trigger

The analog trigger circuitry routing is shown in the [Figure 3-39](#). The analog multiplexer could select either a direct analog input from the EXTATRIG pin (SRC1 in [Figure 3-39](#)) on the 68-pin connector CN1 or the input signal of ADC (SRC2 in [Figure 3-39](#)). That is, the first channel input you fill in the Channel Gain Queue). SRC1 can be used for all trigger modes while SRC2 can only be used for post and delay trigger modes. The range of trigger level for SRC1 is  $\pm 10V$  and the resolution is 78mV (please refer to [Table 3-9](#), the valid code range is from 1 to 255), while the trigger range of SRC2 is the full-scale range of the first channel input in Channel Gain Queue, and



the resolution is the desired range divided by 256. For example, if the first channel input in Channel Gain Queue is CH0 with bipolar  $\pm 5V$  range, the trigger voltage would be 4.96V when the trigger level code is set to 0xFF while  $-4.96V$  when the code is set to 0x01.

Figure 3-39  
**Analog trigger block diagram**

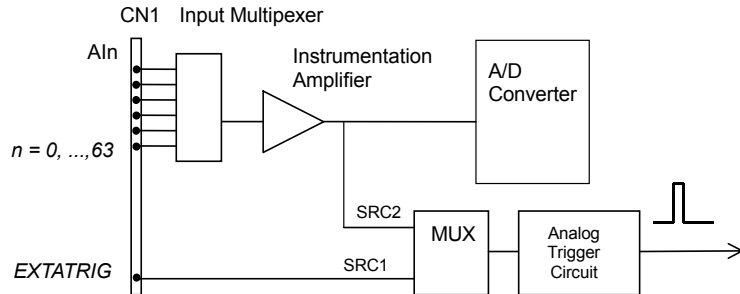


Table 3-9  
**Analog trigger SRC1 (EXTATRIG) ideal transfer characteristic**

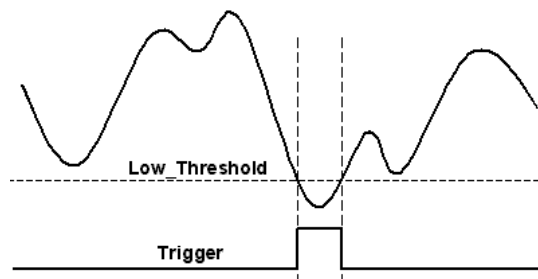
Trigger Level digital setting	Trigger voltage
0xFF	9.92V
0xFE	9.84V
---	---
0x81	0.08V
0x80	0
0x7F	-0.08V
---	---
0x01	-9.92V

The trigger signal is generated when the analog trigger condition is satisfied. There are five analog trigger conditions in KPXI-DAQ modules. KPXI-DAQ modules use two threshold voltages: Low\_Threshold and High\_Threshold to build the five different trigger conditions. Users could configure the trigger conditions easily by software.

**Below-Low analog trigger condition**

Figure 3-40 shows the below-low analog trigger condition, the trigger signal is generated when the input analog signal is less than the Low\_Threshold voltage, and the High\_Threshold setting is not used in this trigger condition.

Figure 3-40  
**Below-Low analog trigger condition**

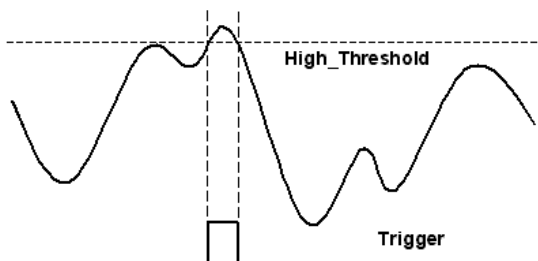


### Above-High analog trigger condition

Figure 3-41 shows the above-high analog trigger condition, the trigger signal is generated when the input analog signal is higher than the High\_Threshold voltage, and the Low\_Threshold setting is not used in this trigger condition.

Figure 3-41

#### Above-High analog trigger condition

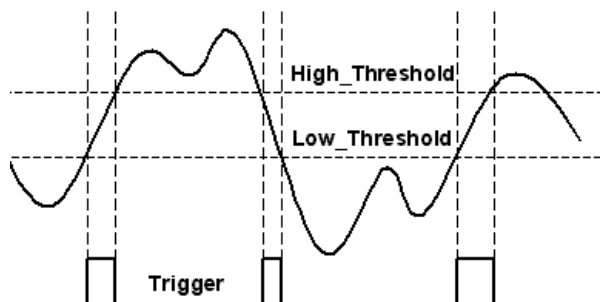


### Inside-Region analog trigger condition

Figure 3-42 shows the inside-region analog trigger condition, the trigger signal is generated when the input analog signal level falls in the range between the High\_Threshold and the Low\_Threshold voltages. The High\_Threshold setting should be always higher than the Low\_Threshold voltage setting.

Figure 3-42

#### Inside-Region analog trigger condition

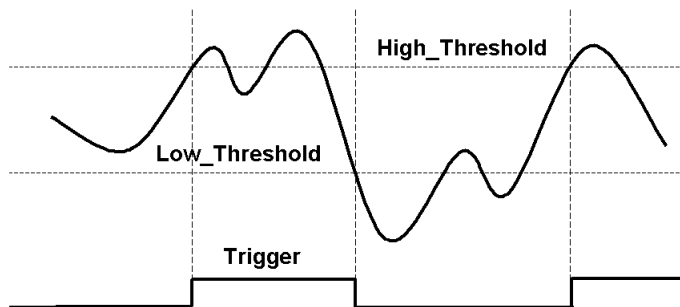


### High-Hysteresis analog trigger condition

Figure 3-43 shows the high-hysteresis analog trigger condition, the trigger signal is generated when the input analog signal level is greater than the High\_Threshold voltage, and the Low\_Threshold voltage determines the hysteresis duration.

**NOTE** The High\_Threshold setting should be always higher than the Low\_Threshold voltage setting.

Figure 3-43  
**High-Hysteresis analog trigger condition**

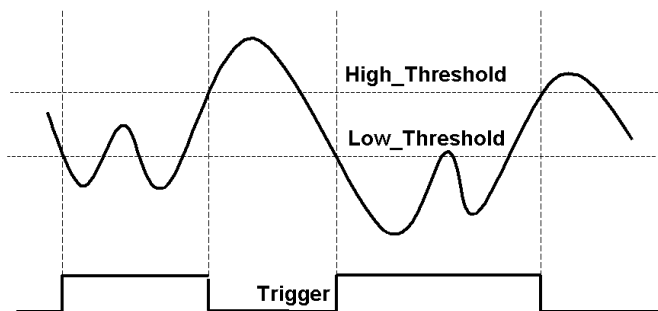


**Low-Hysteresis analog trigger condition**

Figure 3-44 shows the low-hysteresis analog trigger condition, the trigger signal is generated when the input analog signal level is less than the Low\_Threshold voltage, and the High\_Threshold voltage determines the hysteresis duration.

**NOTE** The High\_Threshold setting should be always higher than the Low\_Threshold voltage setting.

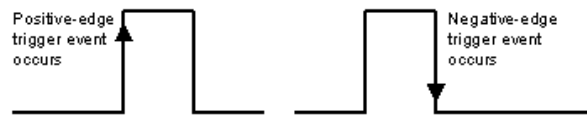
Figure 3-44  
**Low-Hysteresis analog trigger condition**



**External Digital Trigger**

An external digital trigger occurs when a rising edge or a falling edge is detected on the digital signal connected to the EXTDRIG or the EXTWFTRG of the 68-pin connector for external digital trigger. The EXTDRIG is dedicated for A/D process, and the EXTWFTRG is used for D/A process. Users can program the trigger polarity through Keithley Instruments' software drivers easily. Note that the signal level of the external digital trigger signals should be TTL-compatible, and the minimum pulse is 20ns.

Figure 3-45  
External digital trigger



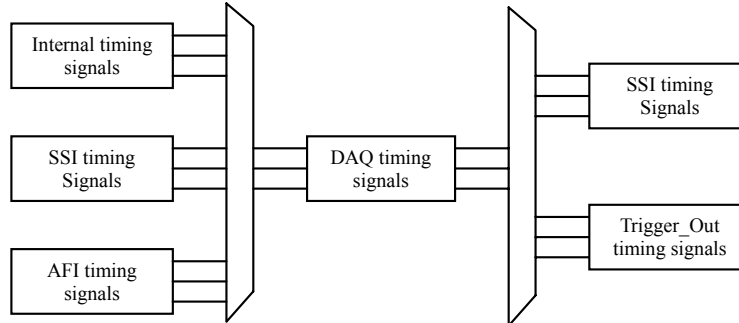
## User-controllable Timing Signals

In order to meet the requirements for user-specific timing and the requirements for synchronizing multiple cards, the KPXI-DAQ series provides flexible user-controllable timing signals to connect to external circuitry or additional cards.

The whole DAQ timing of the KPXI-DAQ series is composed of a bunch of counters and trigger signals in the FPGA. These timing signals are related to the A/D, D/A conversions and Timer/Counter applications. These timing signals can be inputs to or outputs from the I/O connectors or the PXI bus. Therefore the internal timing signals can be used to control external devices or circuitry's operation.

We implemented signal multiplexers in the FPGA to individually choose the desired timing signals for the DAQ operations, as shown in the [Figure 3-46](#).

Figure 3-46  
DAQ signals routing



Users can utilize the flexible timing signals through our software drivers, and simply and correctly connect the signals with the KPXI-DAQ series cards. Here is the summary of the DAQ timing signals and the corresponding functionality for KPXI-DAQ series.

Table 3-10  
Summary of user-controllable timing signals

Timing signal category	Corresponding functionality
SSI/PXI signals	Multiple cards synchronization
AFI signals	Control KPXI-DAQ module by external timing signals

## DAQ timing signals

**NOTE** SCAN\_START, ADCONV and DA\_TRIG, DAWR are supported by KPXI-DAQ-96-3M.

The user-controllable DAQ timing-signals include:

**NOTE** Refer to [Scan Timing and Procedure](#) for the internal timing signal definition.

- TIMEBASE, providing TIMEBASE for all DAQ operations, which could be from internal 40MHz oscillator, EXTTIMEBASE from I/O connector or the SSI\_TIMEBASE. Note that the frequency range of the EXTTIMEBASE is 1MHz to 40MHz, and the EXTTIMEBASE should be TTL-compatible.
- AD\_TRIG, the trigger signal for the A/D operation, which could come from external digital trigger, analog trigger, internal software trigger and SSI\_AD\_TRIG. Refer to [Trigger Sources](#) for detailed description.
- SCAN\_START, the signal to start a scan, which would bring the following ADCONV signals for AD conversion, and could come from the internal SI\_counter, AFI[0] and SSI\_AD\_START. This signal is synchronous to the TIMEBASE. Note that the AFI[0] should be TTL-compatible and the minimum pulse width should be the pulse width of the TIMEBASE to guarantee correct functionality.
- ADCONV, the conversion signal to initiate a single conversion, which could be derived from internal counter, AFI[0] or SSI\_ADCONV. Note that this signal is edge-sensitive. When using AFI[0] as the external ADCONV source, each **rising edge** of AFI[0] would bring an effective conversion signal. Also note that the AFI[0] signal should be TTL-compatible and the minimum pulse width is 20ns.
- DA\_TRIG, the trigger signal for the D/A operation, which could be derived from external digital trigger, analog trigger, internal software trigger and SSI\_AD\_TRIG. Refer to [Trigger Sources](#) for detailed description.
- DAWR, the update signal to initiate a single D/A conversion, which could be derived from internal counter, AFI[1] or SSI\_DAWR. Note that this signal is edge-sensitive. When using AFI[1] as the external DAWR source, each **rising edge** of AFI[1] would bring an effective update signal. Also note that the AFI[1] signal should be TTL-compatible and the minimum pulse width is 20ns.

**NOTE** The System Synchronization Interface (SSI) signals can be routed to the PXI trigger bus for multiple module synchronization within a chassis.

### Auxiliary Function Inputs (AFI)

**NOTE** *EXTWFTRIG and AFI[1] are NOT supported by KPXI-DAQ-96-3M.*

Users could use the AFI in applications that take advantage of external circuitry to directly control the KPXI-DAQ cards. The AFI includes 2 categories of timing signals: one group is the dedicated input, and the other is the multi-function input. [Table 3-11](#) illustrates this categorization.

A summary of the auxiliary function input signals and the corresponding functionality is contained in [Table 3-11](#).

Table 3-11

**Auxiliary function input signals**

Category	Timing signal	Functionality	Constraints
Dedicated input	EXTTIMEBASE	Replace the internal TIMEBASE	TTL-compatible 1MHz to 40MHz Affects on both A/D and D/A operations
	EXTDTRIG	External digital trigger input for A/D operation	TTL-compatible Minimum pulse width = 20ns Rising edge or falling edge
	EXTWFTRG	External digital trigger input for D/A operation	TTL-compatible Minimum pulse width = 20ns Rising edge or falling edge
Multi-function input	AFI[0] (Dual functions)	Replace the internal ADCONV	TTL-compatible Minimum pulse width = 20ns Rising-edge sensitive only
		Replace the internal SCAN_START	TTL-compatible Minimum Pulse width > 2/ TIMEBASE
	AFI[1]	Replace the internal DAWR	TTL-compatible Minimum pulse width = 20ns Rising-edge sensitive only

**EXTDTRIG and EXTWFTRIG**

EXTDTRIG and EXTWFTRIG are dedicated digital trigger input signals for A/D and D/A operations respectively. Please refer to [External Digital Trigger](#) for detailed descriptions.

**EXTTIMEBASE**

When the applications needs specific sampling frequency or update rate that the card could not generate from its internal TIMEBASE, the 40MHz clock, users could utilize the EXTTIMEBASE with internal counters to achieve the specific timing intervals for both A/D and D/A operations. Note that once you choose the TIMEBASE source, both A/D and D/A operations will be affected because A/D and D/A operations share the same TIMEBASE

**AFI[0]**

Alternatively, users can also directly apply an external A/D conversion signal to replace the internal ADCONV signal. This is another way to achieve customized sampling frequencies. The external ADCONV signal can only be inputted from the AFI[0]. As [A/D Conversion](#) describes, the SI\_counter triggers the generation of the A/D conversion signal, ADCONV, but when using the AFI[0] to replace the internal ADCONV signal, then the SI\_counter and the internally generated SCAN\_START will not be effective. By controlling the ADCONV externally, users can sample the data according to external events. In this mode, the Trigger signal and trigger mode settings are not available.

AFI[0] could also be used as SCAN\_START signal for A/D operations. Please refer to [A/D Conversion](#) and [DAQ timing signals](#) for detailed descriptions of the SCAN\_START signal. When using external signal (AFI[0]) to replace the internal SCAN\_START signal, the pulse width of the AFI[0] must be greater than two times the period of Timebase. This feature is suitable for the KPXI series which can scan multiple channels of data controlled by an external event. Note that the AFI[0] is a multi-purpose input, and it can only be utilized for one function at any one time.

**AFI[1]**

Regarding the D/A operations, users could directly input the external D/A update signal to replace the internal DAWR signal. This is another way to achieve customized D/A update rates. The external DAWR signal can only be inputted from the AFI[1]. Note that the AFI[1] is a multi-purpose input, and it can only be utilized for one function at any one time. AFI[1] currently only has one function. Keithley Instruments reserves it for future development.

**System Synchronization Interface**

SSI (System Synchronization Interface) provides the DAQ timing synchronization between multiple cards. In KPXI-DAQ series, we designed a bi-directional SSI I/O to provide flexible connection between cards and allow one SSI master to output the signal and up to three slaves to receive the SSI signal. Note that the SSI signals are designed for card synchronization only, not for external devices.

Table 3-12  
**SSI timing signal summary**

SSI timing signal	Functionality
SSI_TIMEBASE	SSI master: send the TIMEBASE out SSI slave: accept the SSI_TIMEBASE to replace the internal TIMEBASE signal. Note: Affects on both A/D and D/A operations
SSI_AD_TRIG	SSI master: send the internal AD_TRIG out SSI slave: accept the SSI_AD_TRIG as the digital trigger signal.
SSI_ADCONV	SSI master: send the ADCONV out SSI slave: accept the SSI_ADCONV to replace the internal ADCONV signal.
SSI_SCAN_START	SSI master: send the SCAN_START out SSI slave: accept the SSI_SCAN_START to replace the internal SCAN_START signal.
SSI_DA_TRIG	SSI master: send the DA_TRIG out. SSI slave: accept the SSI_DA_TRIG as the digital trigger signal.
SSI_DAWR	SSI master: send the DAWR out. SSI slave: accept the SSI_DAWR to replace the internal DAWR signal.

In PXI form factor, we utilize the PXI trigger bus built on the PXI backplane to provide the necessary timing signal connections. All the SSI signals are routed to the P2 connector. No additional cable is needed. For detailed information of the PXI specifications, please refer to PXI specification Revision 2.0 from PXI System Alliance ([www.pxisa.org](http://www.pxisa.org)).

The 6 internal timing signals could be routed to the PXI trigger bus through software drivers. Please refer to [DAQ timing signals](#) for detailed information of the 6 internal timing signals. Physically the signal routings are accomplished in the FPGA. Cards that are connected together through the SSI or the PXI trigger bus, will still achieve synchronization on the 6 timing signals.

**The mechanism of the SSI/PXI**

1. We adopt master-slave configuration for SSI/PXI. In a system, for each timing signal, there shall be only one master, and other cards are SSI slaves or with the SSI function disabled.
2. For each timing signal, the SSI master doesn't have to be in a single card.

For example:

We want to synchronize the A/D operation through the ADCONV signal for 4 KPXI-DAQ cards. Card 1 is the master, and Card 2, 3, 4 are slaves. Card 1 receives an external digital trigger to start the post trigger mode acquisition. The SSI setting could be:

- Set the SSI\_ADCONV signal of Card 1 to be the master.
- Set the SSI\_ADCONV signals of Card 2, 3, 4 to be the slaves.
- Set external digital trigger for Card 1's A/D operation.
- Set SI\_counter, SI2\_counter, NumChan\_counter and the post scan counter (PSC) on all other cards.
- Start DMA operations for all cards, thus all the cards are waiting for the trigger event.

When the digital trigger condition of Card 1 occurs, Card 1 will internally generate the ADCONV signal and output this ADCONV signal to SSI\_ADCONV signal of Card 2, 3 and 4 through the SSI/PXI connectors. Thus we can achieve simultaneous acquisition across four modules.

You could arbitrarily choose each of the 6 timing signals as the SSI master from any one of the cards. The SSI master can output the internal timing signals to the SSI slaves. With the SSI, users could achieve better card-to-card synchronization.

Note that when power-up or reset, the DAQ timing signals are reset to use the internal generated timing signals.

## Calibration

This section introduces the calibration process to minimize AD measurement errors and DA output errors.

### Loading Calibration Constants

The KPXI-DAQ modules are factory calibrated before shipment by writing the associated calibration constants of TrimDACs to the on-board EEPROM. TrimDACs are devices containing multiple DACs within a single package. TrimDACs do not have memory capability. That means the calibration constants do not retain their values after the system power is turned off. Loading calibration constants is the process of loading the values of TrimDACs stored in the on-board EEPROM. Keithley Instruments provides software to make it easy to read the calibration constants automatically when necessary.

There is a dedicated space for calibration constants in the EEPROM. In addition to the default bank of factory calibration constants, there are three extra user-modifiable banks. This means users can load the TrimDACs values either from the original factory calibration or from a calibration that is subsequently performed.

Because of the fact that errors in measurements and outputs will vary with time and temperature, it is recommended to conduct re-calibration when the card is installed in the users environment. The auto-calibration function used to minimize errors will be introduced in the next sub-section.

### Auto-calibration

By using the auto-calibration feature of the KPXI-DAQ module, the calibration software can measure and correct almost all the calibration errors without any external signal connections, reference voltages, or measurement devices.

The KPXI-DAQ module has an on-board calibration reference to ensure the accuracy of auto-calibration. The reference voltage is measured at the factory and adjusted through a digital potentiometer by using an ultra-precision calibrator. The impedance of the digital potentiometer is memorized after this adjustment. It is not recommended for users to adjust the on-board calibration reference except when an ultra-precision calibrator is available.



- NOTE** 1. Before auto-calibration procedure starts, it is recommended to warm up the card for at least 15 minutes.
2. Please remove the cable before an auto-calibration procedure is initiated because the DA outputs would be changed in the process of calibration.

## **Saving Calibration Constants**

After an auto-calibration is completed, users can save the new calibration constants into one of the three user-modifiable banks in the EEPROM. The date and the temperature when you ran the auto-calibration will be saved accompanied with the calibration constants. This means users can store three sets of calibration constants according to three different environments and re-load the calibration constants later.

This page left blank intentionally.

**In this appendix:**

Topic	Page
Introduction to KDAQ-DRVR.....	A-2
About the KDAQ-DRVR software .....	A-2
KDAQ-DRVR hardware support .....	A-2
KDAQ-DRVR language support .....	A-2
Fundamentals of building applications with KDAQ-DRVR.....	A-3
Microsoft® Visual Basic (Version 6.0).....	A-3
Using Microsoft Visual Basic.NET .....	A-4
Microsoft Visual C/C++ .....	A-4
KDAQ-DRVR utilities for Win32 .....	A-5
KDAQ-DRVR configuration utility (configdrv) .....	A-5
KDAQ-DRVR data file converter utility (KiDAQCvt).....	A-6
KDAQ-DRVR overview .....	A-6
General configuration function group .....	A-7
Analog input function group .....	A-7
Analog output function group.....	A-10
Digital input function group .....	A-12
Digital output function group .....	A-13
General timer/counter function group .....	A-13
DIO function group .....	A-13
SSI function group .....	A-14
Calibration function group.....	A-14
KDAQ-DRVR application hints.....	A-15
Analog input programming hints .....	A-16
Analog output programming hints .....	A-36
One-shot analog output programming scheme .....	A-36
Digital input programming hints .....	A-50
Digital output programming hints .....	A-51
DAQ event message programming hints.....	A-52
Continuous data transfer in KDAQ-DRVR .....	A-53
Continuous data transfer mechanism .....	A-53
Double-buffered AI/AO operation .....	A-53
Single-buffered versus double-buffered data transfer .....	A-54
Pre-trigger mode/middle-trigger data acquisition (AI) .....	A-54

## Introduction to KDAQ-DRVR

### About the KDAQ-DRVR software

KDAQ-DRVR is a software development kit for Keithley Instruments KPXI-DAQ cards. It contains a high performance data acquisition driver for developing custom applications under Microsoft Windows® XP/2000 environments.

KDAQ-DRVR was developed to provide a simple programming interface in communication with the Keithley Instruments' KPXI-DAQ Series cards. The KDAQ-DRVR's easy-to-use memory and data-buffer management capabilities frees developers from these issues but still allows high-level access to the card's features.

Using KDAQ-DRVR also takes advantage of the power and features of Win32® System for data acquisition applications, including the ability to run multiple applications and utilize extended memory. Using KDAQ-DRVR in a Visual Basic® environment makes it easy to create custom user interfaces and graphics.

In addition to the software drivers, sample programs are provided for your reference. These sample programs will save programming time and highlight some of the KPXI-DAQ Series cards features.

### KDAQ-DRVR hardware support

Keithley Instruments will periodically upgrade the KDAQ-DRVR for new KPXI-DAQ cards. Refer to the Release Notes for the most recent list of cards that the current KDAQ-DRVR supports. The following cards are supported by KDAQ-DRVR:

- **KPXI-SDAQ-4-2M**: 2MHz 4 channels simultaneous A/D and 2 channels D/A output device with bus mastering DMA transfer capability
- **KPXI-SDAQ-4-500K**: 500kHz 4 channels simultaneous A/D and 2 channels D/A output device with bus mastering DMA transfer capability
- **KPXI-DAQ-64-3M**: 3MHz 64 channels multiplexed A/D and 2 channels D/A output device with bus mastering DMA transfer capability
- **KPXI-DAQ-64-500K**: 500kHz 64 channels multiplexed A/D and 2 channels D/A output device with bus mastering DMA transfer capability
- **KPXI-DAQ-64-250K**: 250kHz 64 channels multiplexed A/D and 2 channels D/A output device with bus mastering DMA transfer capability
- **KPXI-DAQ-96-3M**: 3MHz 96 channels multiplexed A/D device with bus mastering DMA transfer capability
- **KPXI-AO-4-1M**: High performance, 4 channels analog output, multi-function device with bus mastering DMA transfer capability
- **KPXI-AO-8-1M**: High performance, 8 channels analog output, multi-function device with bus mastering DMA transfer capability

### KDAQ-DRVR language support

KDAQ-DRVR is the DLL (Dynamic-Link Library) version for use with Windows XP/2000. It works with any Windows programming language that allows calls to a DLL, such as Microsoft Visual C/C++ (4.0 or above), Borland C++ (5.0 or above), or Microsoft Visual Basic (4.0 or above).

## Fundamentals of building applications with KDAQ-DRVR

The following paragraphs outline how to create Windows KDAQ-DRVR projects using Microsoft® Visual Basic® (Version 6.0), Microsoft Visual Basic.NET, and Microsoft Visual C/C++.

### Microsoft® Visual Basic (Version 6.0)

To create a Windows® XP/2000 Keithley KDAQ-DRVR application using the API and Microsoft Visual Basic, follow these steps:

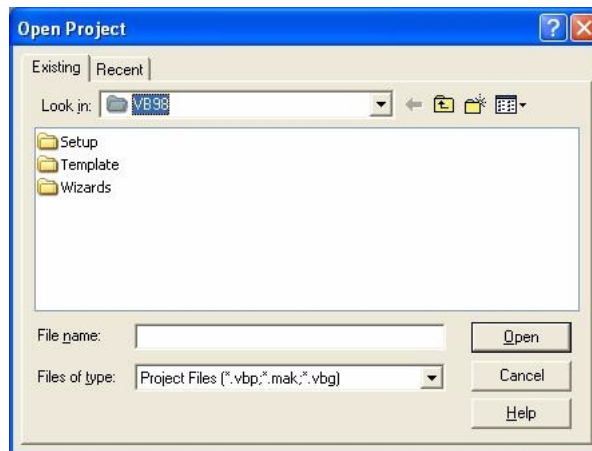
#### Step 1: Enter Visual Basic and open or create a project to use KDAQ-DRVR

To create a new project, select **New Project** from the **File** menu.

To use an existing project:

1. Open the file by selecting **Open Project** from the **File** menu. The **Open Project** dialog box appears (Figure A-1).

Figure A-1  
Open Project dialog box



2. Load the project by finding and double-clicking the project file name in the applicable directory.

#### Step 2: Include function declarations and constants file (KDAQDRVR.BAS)

If it is not already included in the project, add the **KDAQDRVR.BAS** file as a module to your project. All function declarations and constants are contained in this file. These function declarations and constants are used to develop user data acquisition applications.

#### Step 3: Design the application interface

Add elements, such as a command button, list box, or text box, etc., on the Visual Basic form used to design the interface. These elements are standard controls from the Visual Basic Toolbox. To place a needed control on the form:

1. Select the needed control from the **Toolbox**.
2. Draw the control on the form. Alternatively, to place the default-sized control on the form, click the form. Use the **Select Objects** tool to reposition or resize controls.

### Step 4: Set control properties

Set control properties from the properties list. To view the properties list, select the desired control and do one of the following:

- Press **F4**
  - Select the **Properties** command in the **View** menu
- or
- Click the **Properties** button on the Toolbar.

### Step 5: Write the event codes

The event codes define the action desired when an event occurs. To write the event codes:

1. Double-click the control or form needing event code (the code module will appear).
2. Add new code as needed. All functions that are declared in **KDAQDRVR.BAS** can be called to perform data acquisition operations (refer to tables contained later in this manual).

### Step 6: Run your application

To run the application, either:

- Press **F5**
  - Select **Start** from the **Run** menu
- or
- Click the **Start** icon on the Toolbar

## Using Microsoft Visual Basic.NET

To create a data acquisition application using KDAQ-DRVR and Visual Basic.NET, use the procedure for [Microsoft® Visual Basic \(Version 6.0\)](#) as an outline, but in step 2, use the file named **KDAQDRVR.VB** (instead of the file named **KDAQDRVR.BAS**).

## Microsoft Visual C/C++

To create a Windows XP/2000 KDAQ-DRVR library application using the KDAQ-DRVR function library and Microsoft Visual C/C++, follow these steps:

### Step 1: Enter Visual C/C++ and open or create a project that will use the KDAQ-DRVR

*NOTE* The project can be a new or existing one.

### Step 2: Include function declarations and constants file (KDAQDRVR.H)

Include **KDAQDRVR.H** in the C/C++ source files that call KDAQ-DRVR functions by adding the following statement in the source file:

```
#include "kdaqdrvr.h"
```

*NOTE:* KDAQ-DRVR function declarations and constants are contained in **kdaqdrvr.h**. Use the functions and constants to develop user-self data acquisition applications.

### Step 3: Build your application

1. Set suitable compile and link options.

2. Select **Build** from the **Build** menu (Visual C/C++ 4.0 and higher).
3. Remember to link the Keithley Command Compatible library: **KDAQ-DRVR.LIB**

## KDAQ-DRVR utilities for Win32

The following information describes the tools that accompany the KDAQ-DRVR package.

### KDAQ-DRVR configuration utility (configdrv)

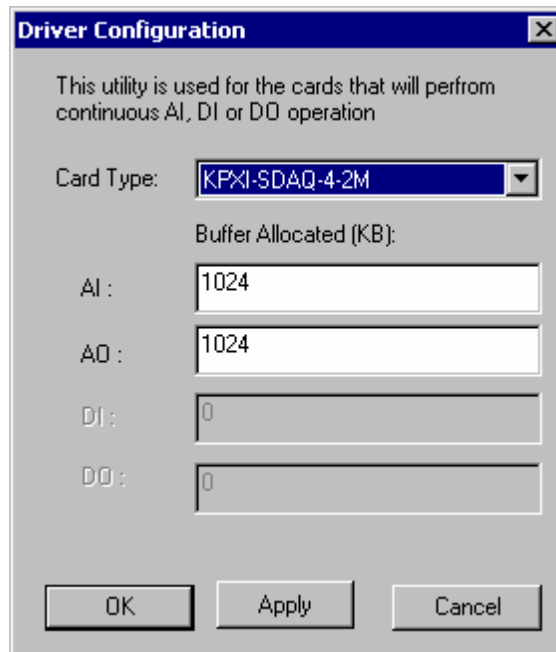
The **configdrv** utility sets or changes the allocated buffer sizes of AI (analog input), AO (analog output), DI (digital input) and DO (digital output). The default location of this utility is in the **<InstallDir>\Util** directory.

The allocated buffer sizes (of AI, AO, DI, DO) represent the amount of memory assigned to each buffer. The memory is allocated in page KB (1024 bytes per page). The device driver will try to allocate the required size of memory at system startup. The size of the initial allocated memory is the maximum memory size that DMA (Direct Memory Access) or interrupt transfer can use. If the memory required exceeds the initial allocated size, an unexpected result in that transfer (DMA or interrupt) will result.

The **Driver Configuration** window is shown in [Figure A-2](#). To change the allocated buffer settings for a single KDAQ-DRVR driver, use the **configdrv** utility and select the model number of the card from the **Card type: drop-down menu** (for example, KPXI-SDAQ-4-2).

The allocated buffer size fields of AI, AO, DI and DO are the default (or previously set) values. To change a value, type the desired value in the box corresponding to AI, AO, DI, or DO. Set the value according to the requirements of your applications. Click **Apply** and then **OK** to finish changing the settings.

Figure A-2  
Driver Configuration window



## KDAQ-DRVR data file converter utility (KiDAQCvt)

Data files generated by KDAQ-DRVR functions which perform continuous data acquisition and storage operations, are written to disk in binary format. Since a binary file cannot be read by either a normal text editor or accessed by Microsoft Excel<sup>®</sup>, KDAQ-DRVR provides a tool named **KiDAQCvt** to convert the binary file to a file format easily read. The default location of this utility is **<InstallDir>\Util** directory.

The **KiDAQCvt** main window includes two frames: the upper frame (called the **Input File** frame) is used for the source data file; the lower frame which is used for the destination file.

To load the source binary data file, type the binary data file name in **File Path** field or click **Browse** to select the source file from the **Input File** frame. Click the **Load** button to process the selected file. As the file is loaded, the information related to the data file (e.g., data type, data width, AD Range, etc.) are shown in the corresponding fields in the **Input File** frame. The default converted data file path and format are also listed.

The default destination file has a **.cvt** extension and is located in the same directory as the source. To change the default setting, type the desired file path or browse from **Output File** frame to the desired destination file location.

**KiDAQCvt** provides three types of data format conversion: [Text file with scaled data](#), [Binary file with scaled data](#), and [Text file with binary codes](#).

### Text file with scaled data

Data in hexadecimal format is scaled to the engineering unit (voltage, amplitude, etc.) according to the card type, data width, and data range. Then, it is written to disk in *text* file format. This type of data is available for data accessed from continuous Analog Input (AI) operation only.

### Binary file with scaled data

Data in hexadecimal format is scaled to the engineering unit (voltage, amplitude, etc.) according to the card type, data width, and data range. Then, it is written to disk in *binary* file format. This type of data is available using continuous Analog Input (AI) operation only.

### Text file with binary codes

Data in hexadecimal format or converted to a decimal value is written to disk in text file format. If the original data includes channel information, the raw value will be handled to get the real data value. This type of data is available using continuous AI and DI operations.

The text delimiter in the converted file is user-selectable between space, comma, and tab.

To add title/head (which includes the card type information at the beginning of the file), check the **Title/Head** box.

After setting the properties (file path, format, etc.) related to the converted file, push the **Start Convert** button on the **Output File** frame to perform the file conversion.

## KDAQ-DRVR overview

This overview describes classes of functions in the KDAQ-DRVR. KDAQ-DRVR functions are grouped to the following classes:

- **General configuration function group**
- **Analog input function group**
  - Analog input configuration functions
  - One-shot analog input functions



- Continuous analog input functions
- Asynchronous analog input monitoring functions
- **Analog output function group**
  - Analog output configuration functions
  - One-shot analog output functions
  - Continuous analog output functions
  - Asynchronous analog output monitoring functions
- **Digital input function group**
  - One-shot digital input functions
- **Digital output function group**
  - One-shot digital output functions
- **General timer/counter function group**
- **DIO function group**
  - Digital input/output configuration function
- **SSI function group**
- **Calibration function group**

## General configuration function group

Use these functions to initialize and configure the data acquisition card.

### **KDAQ\_Register\_Card**

Initializes the hardware and software states of a Keithley Instruments PXI DAQ card. Call **Register\_Card** before calling any other KDAQ-DRVR library functions.

### **KDAQ\_Release\_Card**

Tells KDAQ-DRVR library that this registered card is not used currently and can be released. This would make room for a new card to register.

### **KDAQ\_AIO\_Config**

Informs KDAQ-DRVR library of Timer source, and analog trigger source for the Keithley Instruments PXI DAQ Device.

## Analog input function group

### Analog input configuration functions

#### **KDAQ\_AI\_CH\_Config**

Informs KDAQ-DRVR library of the AI range selected for the specified analog input channel of Keithley Instruments PXI DAQ Device. **KDAQ\_AI\_CH\_Config** must be called before calling any function to perform an analog input operation.

#### **KDAQ\_AI\_Config**

Informs KDAQ-DRVR library of trigger source, trigger mode, input mode and trigger properties for the analog input operation of Keithley Instruments PXI DAQ device. **KDAQ\_AI\_Config** must be called before calling any function to perform continuous analog input operation.

**KDAQ\_AI\_MuxScanSetup**

Informs and stores **numChans**, **chans**, and **gain\_refGnd** in the Channel-Gain Queue for a scanned data acquisition operation.

**KDAQ\_AI\_InitialMemoryAllocated**

Gets the actual size of analog input memory that is available in the device driver.

**One-shot analog input functions****KDAQ\_AI\_ReadChannel**

Performs a software triggered A/D conversion (analog input) on an analog input channel and returns the value converted (un-scaled).

**KDAQ\_AI\_SimuReadChannel**

Performs a software triggered A/D conversion (analog input) on analog input channels and returns the values converted (un-scaled). This function is only available for Simultaneous AD card (for example, Keithley Instruments KPXI-SDAQ-4-2M).

**KDAQ\_AI\_ReadMuxScan**

Returns readings for all analog input channels selected by **KDAQ\_AI\_MuxScanSetup**. This function is only available for the Multiplexed AD card (e.g., KPXI-DAQ-64-500K).

**KDAQ\_AI\_ScanReadChannels**

Performs software triggered A/D conversions (analog input) on analog input channels and returns the values converted (un-scaled). This function is only available for the Multiplexed AD card (e.g., KPXI-DAQ-64-500K).

**KDAQ\_AI\_VReadChannel**

Performs a software triggered A/D conversion (analog input) on an analog input channel and returns the value scaled to a voltage in units of volts.

**KDAQ\_AI\_VoltScale**

Converts the result from an **KDAQ\_AI\_ReadChannel** call to the actual input voltage.

**Continuous Analog Input functions****KDAQ\_AI\_ContReadChannel**

On the specified analog input channel, this function performs continuous A/D conversions at a rate that is as close as possible to the rate specified.

**KDAQ\_AI\_ContScanChannels**

Performs continuous A/D conversions on the specified continuous analog input channels at an available rate closest to the rate you specified. This function is only available for those cards that support auto-scan functionality.

**KDAQ\_AI\_ContReadMultiChannels**

On the specified analog input channels, this function performs continuous A/D conversions at the closest available rate to the rate specified. This function is only available for those cards that support auto-scan functionality.

**KDAQ\_AI\_ContReadChannelToFile**

On the specified analog input channel, this function performs continuous A/D conversions at an available rate that is closest to the rate specified and saves the acquired data in a disk file.

**KDAQ\_AI\_ContScanChannelsToFile**

On the specified continuous analog input channels, this function performs continuous A/D conversions at the available rate that is closest to the rate specified and saves the acquired data in a disk file. This function is only available for those cards that support auto-scan functionality.

**KDAQ\_AI\_ContReadMultiChannelsToFile**

On the specified analog input channels, this function performs continuous A/D conversions at the available rate closest to the rate specified and saves the acquired data in a disk file. This function is only available for those cards that support auto-scan functionality.

**KDAQ\_AI\_ContMuxScan**

This function initializes the Channel-Gain Queue to point to the start of the scan sequence as specified by `KDAQ_AI_MuxScanSetup` and starts a multiple-channel scanned data acquisition operation. This function is only available for the Multiplexed AD card (e.g. KPXI-DAQ-64-500K).

**KDAQ\_AI\_ContMuxScanToFile**

Initializes the Channel-Gain Queue to point to the start of the scan sequence as specified by `KDAQ_AI_MuxScanSetup`, starts a multiple-channel scanned data acquisition operation, and saves the acquired data in a disk file.

**KDAQ\_AI\_ContVScale**

Converts the values of an array of acquired data from a continuous A/D conversion call to the actual input voltages.

**KDAQ\_AI\_ContStatus**

Checks the current status of the continuous analog input operation.

**KDAQ\_AI\_EventCallback**

Controls and notifies the user's application when a specified DAQ event occurs. The notification is performed through a user-specified callback function.

**KDAQ\_AI\_ContBufferSetup**

Sets up the buffer for continuous analog input.

**KDAQ\_AI\_ContBufferReset**

Resets all buffers set by function `KDAQ_AI_ContBufferSetup`.

**Asynchronous analog input monitoring functions****KDAQ\_AI\_AsyncCheck**

Checks the current status of the asynchronous analog input operation.

**KDAQ\_AI\_AsyncClear**

Stops the asynchronous analog input operation.

**KDAQ\_AI\_AsyncDbfBufferMode**

Enables or Disables double buffer data acquisition mode.

**KDAQ\_AI\_AsyncDbfBufferHalfReady**

Checks whether the next half buffer of data in the circular buffer is ready for transfer during an asynchronous double-buffered analog input operation.

**KDAQ\_AI\_AsyncDbfBufferToFile**

Copies half of the data of the circular buffer into a disk file.

**KDAQ\_AI\_AsyncDbfBufferOverrun**

Checks or clears overrun status of the double-buffered analog input operation.

**KDAQ\_AI\_AsyncDbfBufferHandled**

Notifies KDAQ-DRVR the ready buffer has been handled in a user application.

**KDAQ\_AI\_AsyncReTrigNextReady**

Checks whether the data associated to the next trigger signal is ready during an asynchronous re-triggered analog input operation.

## Analog output function group

### Analog output configuration functions

**KDAQ\_AO\_CH\_Config**

Informs KDAQ-DRVR library of the reference voltage value selected for an analog output channel of a Keithley Instruments PXI DAQ Device. **KDAQ\_AO\_CH\_Config** must be called before calling the function to perform the voltage output operation.

**KDAQ\_AO\_Config**

Informs KDAQ-DRVR library of trigger source, trigger mode, output mode and trigger properties for the analog output operation of Keithley Instruments PXI DAQ Device. **KDAQ\_AO\_Config** must be called before calling the function to perform continuous analog output operation of the Keithley Instruments PXI DAQ Device.

**KDAQ\_AO\_InitialMemoryAllocated**

Gets the actual size of analog output DMA memory that is available in the device driver.

**KDAQ\_AO\_Group\_Setup**

Assigns one or more analog output channels to a waveform generation group.

**KDAQ\_AO\_Group\_WFM\_StopConfig**

Informs KDAQ-DRVR library of stop source and stop mode for the asynchronous analog output operation of a specified group.

## One-shot analog output functions

### **KDAQ\_AO\_WriteChannel**

Writes a binary value to the specified analog output channel.

### **KDAQ\_AO\_SimuWriteChannel**

Writes binary values to the specified analog output channels simultaneously. This function is only available for Simultaneous DA card.

### **KDAQ\_AO\_VWriteChannel**

Accepts a voltage value, scales it to the proper binary value and writes a binary value to the specified analog output channel.

### **KDAQ\_AO\_VoltScale**

Scales a voltage to a binary value.

### **KDAQ\_AO\_Group\_Update**

Writes binary values to the specified group of analog output channels simultaneously.

### **KDAQ\_AO\_Group\_VUpdate**

Accepts voltage values, scales them to the proper binary values and writes binary values to the specified group of analog output channels simultaneously.

## Continuous analog output functions

### **KDAQ\_AO\_ContWriteChannel**

On the specified analog output port, this function performs continuous analog output at a rate as close as possible to the rate specified.

### **KDAQ\_AO\_ContWriteMultiChannels**

Performs continuous D/A conversions on the specified analog output channels at a rate that is as close as possible to the rate specified.

### **KDAQ\_AO\_ContStatus**

Checks the current status of the continuous analog output operation.

### **KDAQ\_AO\_EventCallback**

Controls and notifies the user's application when a specified DAQ event occurs. The notification is performed through a user-specified callback function.

### **KDAQ\_AO\_ContBufferSetup**

This function sets up the buffer for continuous analog output.

### **KDAQ\_AO\_ContReset**

This function resets all buffers set by function KDAQ\_AO\_ContBufferSetup for continuous analog output.

**KDAQ\_AO\_ContBufferCompose**

This function organizes the data for each channel and fills them in the buffer for continuous analog output operation.

**KDAQ\_AO\_ContBufferComposeAll**

Fills the data for a specified channel in the buffer for continuous analog output operation.

**KDAQ\_AO\_Group\_FIFOLoad**

Loads a waveform buffer to on-board DA FIFOs.

**KDAQ\_AO\_Group\_WFM\_Start**

On the specified group of analog output channels, this function performs continuous D/A conversions at a rate that is as close as possible to the rate specified.

**Asynchronous analog output monitoring function****KDAQ\_AO\_AsyncCheck**

Checks the current status of the asynchronous analog output operation.

**KDAQ\_AO\_AsyncClear**

Stops the asynchronous analog output operation.

**KDAQ\_AO\_AsyncDbIBufferMode**

Enables or Disables double buffer data acquisition mode.

**KDAQ\_AO\_AsyncDbIBufferHalfReady**

Checks whether the next half buffer of data in circular buffer is ready during an asynchronous double-buffered analog output operation.

**KDAQ\_AO\_Group\_WFM\_AsyncCheck**

Checks the current status of the asynchronous analog output operation of a specified group.

**KDAQ\_AO\_Group\_WFM\_AsyncClear**

Stops the asynchronous analog output operation of a specified group.

**Digital input function group****One-shot digital input functions****KDAQ\_DI\_ReadLine**

Reads the digital logic state of the specified digital line in the specified port.

**KDAQ\_DI\_ReadPort**

Reads digital data from the specified digital input port.

## Digital output function group

### One-Shot Digital Output functions

#### **KDAQ\_DO\_WriteLine**

Sets the digital output line in the digital output port to the specified state. This function is only available for those cards that support digital output read-back functionality.

#### **KDAQ\_DO\_WritePort**

Writes digital data to the specified digital output port.

#### **KDAQ\_DO\_ReadLine**

Reads the specified digital output line in the specified digital output port.

#### **KDAQ\_DO\_ReadPort**

Reads digital data from the specified digital output port.

## General timer/counter function group

#### **KDAQ\_GCTR\_Setup**

Controls the general-purpose counter to operate in the specified mode.

#### **KDAQ\_GCTR\_Read**

Reads the counter value of the general-purpose counter without disturbing the counting process.

#### **KDAQ\_GCTR\_Control**

Controls the selected counter/timer by software.

#### **KDAQ\_GCTR\_Reset**

Halts the specified general-purpose timer/counter operation and reloads the initial value of the timer/counter.

#### **KDAQ\_GCTR\_Status**

Reads the counter value of the general-purpose counter without disturbing the counting process.

## DIO function group

### Digital input/output configuration function

#### **KDAQ\_DIO\_PortConfig**

This function is only used by the Digital I/O cards whose I/O port can be set as an input port or output port. This function informs KDAQ-DRVR library of the port direction selected for the digital input/output operation. Call **KDAQ\_DIO\_PortConfig** before calling functions to perform digital input/output operation.

## SSI function group

### **KDAQ\_SSI\_SourceConn**

Connects a device to the specified SSI bus trigger line.

### **KDAQ\_SSI\_SourceDisConn**

Disconnects a device signal from the specified SSI bus trigger line.

### **KDAQ\_SSI\_SourceClear**

Disconnects a device signal from the specified SSI bus trigger line.

## Calibration function group

### **KDAQ\_DB\_Auto\_Calibration\_ALL**

Calibrates your Keithley Instruments PXI DAQ Device.

### **KDAQ\_EEPROM\_CAL\_Constant\_Update**

Save new calibration constants to the specified bank of EEPROM.

### **KDAQ\_Load\_CAL\_Data**

Load calibration constants from the specified bank of EEPROM.

### **SDAQ4M2\_Acquire\_AD\_Error**

Acquires the offset and gain errors of the specified AI channel in the specified polarity mode.

### **SDAQ4M2\_Acquire\_DA\_Error**

Acquires the offset and gain errors of the specified DA channel in the specified polarity mode.

### **SDAQ4K500\_Acquire\_AD\_Error**

Acquires the offset and gain errors of the specified AI channel in the specified polarity mode.

### **SDAQ4K500\_Acquire\_DA\_Error**

Acquires the offset and gain errors of the specified DA channel in the specified polarity mode.

### **DAQ64M3\_Acquire\_AD\_Error**

Acquires the offset and gain errors of ADC.

### **DAQ64M3\_Acquire\_DA\_Error**

Acquires the offset and gain errors of the specified DA channel in the specified polarity mode.

### **DAQ64K500\_Acquire\_AD\_Error**

Acquires the offset and gain errors of ADC.

### **DAQ64K500\_Acquire\_DA\_Error**

Acquires the offset and gain errors of the specified DA channel in the specified polarity mode.



**DAQ64K250\_Acquire\_AD\_Error**

Acquires the offset and gain errors of ADC.

**DAQ64K250\_Acquire\_DA\_Error**

Acquires the offset and gain errors of the specified DA channel in the specified polarity mode.

**DAQ96M3\_Acquire\_AD\_Error**

Acquires the offset and gain errors of ADC.

**AOxM1\_Acquire\_DA\_Error**

Acquires the offset and gain errors of the specified DA channel in the specified polarity mode.

**AOxM1\_Acquire\_AD\_Error**

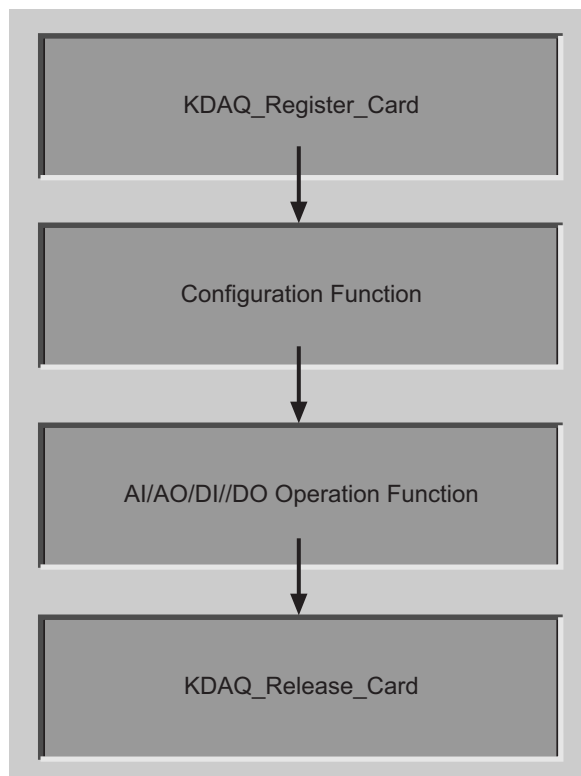
Acquires the offset and gain errors of ADC.

**KDAQ-DRVR application hints**

This paragraph provides the programming function flow that KDAQ-DRVR performs during analog I/O and digital I/O.

The figure below shows the basic building blocks of a KDAQ-DRVR application. Each building block of KDAQ-DRVR uses **KDAQ\_Register\_Card** at the beginning and **KDAQ\_Release\_Card** at the end. Other than that similarity, the functions comprising each building block vary dependent on the specific devices and applications.

Figure A-3  
**KDAQ-DRVR application building blocks**



The programming schemes for analog input/output and digital input/output are described individually in the following sections.

## Analog input programming hints

KDAQ-DRVR provides two kinds of analog input operation: non-buffered single-point analog input readings and buffered continuous analog input operation.

The non-buffered single-point AI uses a software polling method to read data from the device. The programming scheme for this kind of AI operation is described in the paragraph titled [One-shot analog input programming scheme](#).

The buffered continuous analog input uses DMA transfer to send data from the device to user's buffer. The maximum number of counts in one transfer depends on the size of initially allocated memory for the analog input in the driver. We recommend having your applications use the **KDAQ\_AI\_InitialMemoryAllocated** function to get the initial allocated memory size before performing continuous AI operation.

The buffered continuous analog input includes:

- synchronous continuous AI
- non-double-buffered asynchronous continuous AI
- double-buffered asynchronous continuous AI
- pre/middle triggered non-double-buffered asynchronous continuous AI
- pre/middle triggered double-buffered asynchronous continuous AI

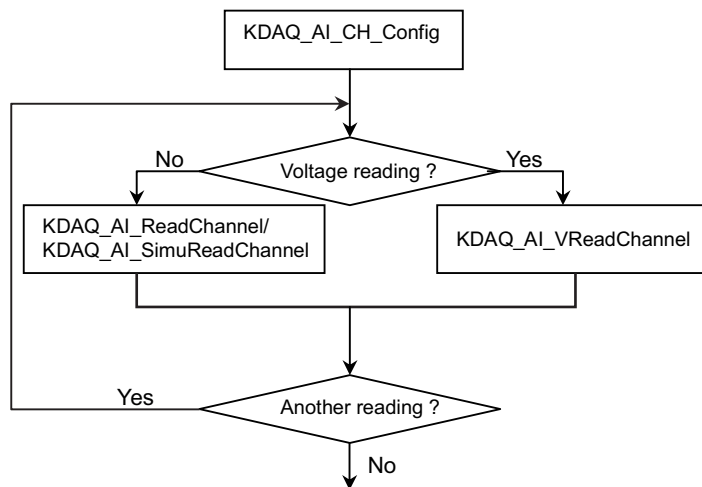
For more information regarding the special consideration and performance issues for the buffered continuous analog input, refer to the Continuous Data Transfer in the KDAQ-DRVR chapter for details.

### One-shot analog input programming scheme

This section describes the typical flow of non-buffered single-point analog input readings.

Figure A-4

#### Typical function flow for all types of KDAQ-DRVR series

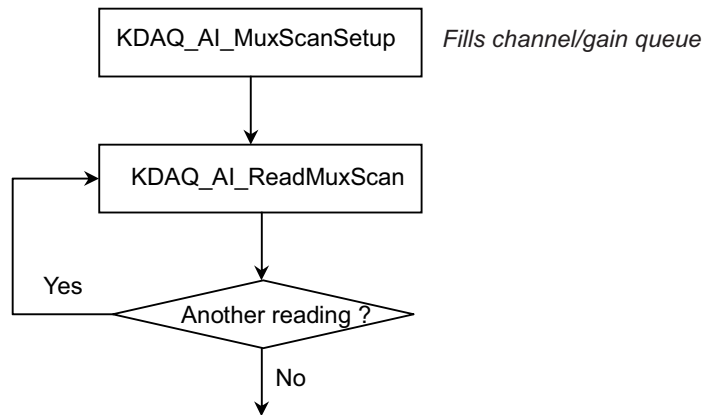


**Example code fragment**

```
card = KDAQ_Register_Card(KPXI_SDAQ_4_2M, card_number);
...
KDAQ_AI_CH_Config (card, channelNo, AD_B_10_V);
KDAQ_AI_ReadChannel(card, channelNo, &analog_input[i]);
...
KDAQ_Release_Card(card);
```

**NOTE:** *Figure A-5 applies to the following cards: KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, and KPXI-DAQ-96-3M.*

Figure A-5  
**Fills channel gain queue first**



**Example code fragment**

```
card = KDAQ_Register_Card(KPXI_DAQ_64_3M, card_number);
CHANNELCOUNT = 1;
chans[0] = 0;
ranges[0] = AD_B_10_V | AI_RSE;
KDAQ_AI_MuxScanSetup(card, CHANNELCOUNT, chans, ranges);
KDAQ_AI_ReadMuxScan (card, chan_data);
...
KDAQ_Release_Card(card);
```

**Continuous analog input (with initial default settings) programming scheme**

This programming section describes the typical flow of synchronous analog input operation performed by the device in a default configuration. For synchronous AI, the **SyncMode** argument in continuous AI functions has to be set as SYNCH\_OP and for asynchronous AI, the **SyncMode** argument has to be set as ASYNCH\_OP.

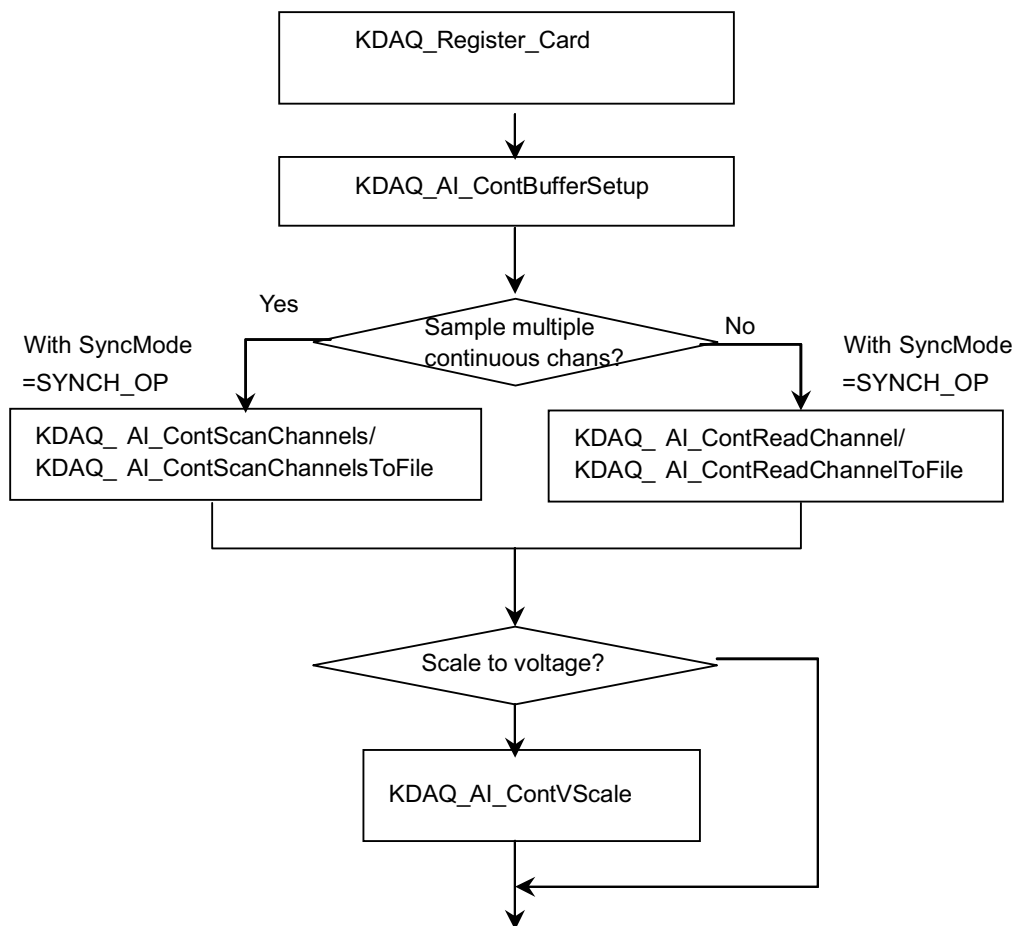
Table A-1  
Initial default channel configuration

Description	Value
AD data range	AD_B_10_V
Reference ground*	AI_RSE
* Reference ground is available for KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K and KPXI-DAQ-96-3M.	

Table A-2  
Initial default AI configuration

Description	Value
A/D conversion source	KDAQ_AI_ADCONVSRC_Int (internal timer pacer)
A/D trigger mode	KDAQ_AI_TRGMOD_POST (post trigger)
A/D trigger source	KDAQ_AI_TRGSRC_SOFT (software trigger)
Auto buffer reset	TRUE

Figure A-6  
Synchronous operation

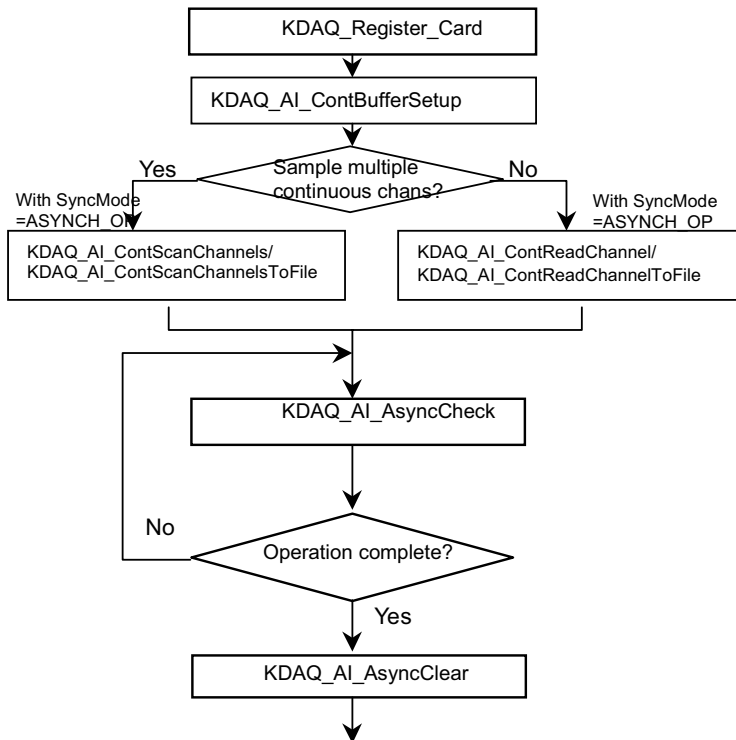


**Example code fragment**

```

card = KDAQ_Register_Card(KPXI_SDAQ_4_2M, card_number);
KDAQ_AI_ContBufferSetup (card, ai_buf, data_size, &Id);
KDAQ_AI_ContScanChannels (card, channel, Id, data_size/(channel+1), scan_intrv,
samp_intrv, SYNCH_OP); or
KDAQ_AI_ContReadChannel(card, channel, Id, data_size, scan_intrv, samp_intrv,
SYNCH_OP)
...
KDAQ_Release_Card(card);
    
```

**Figure A-7**  
**Non-double buffered asynchronous operation**



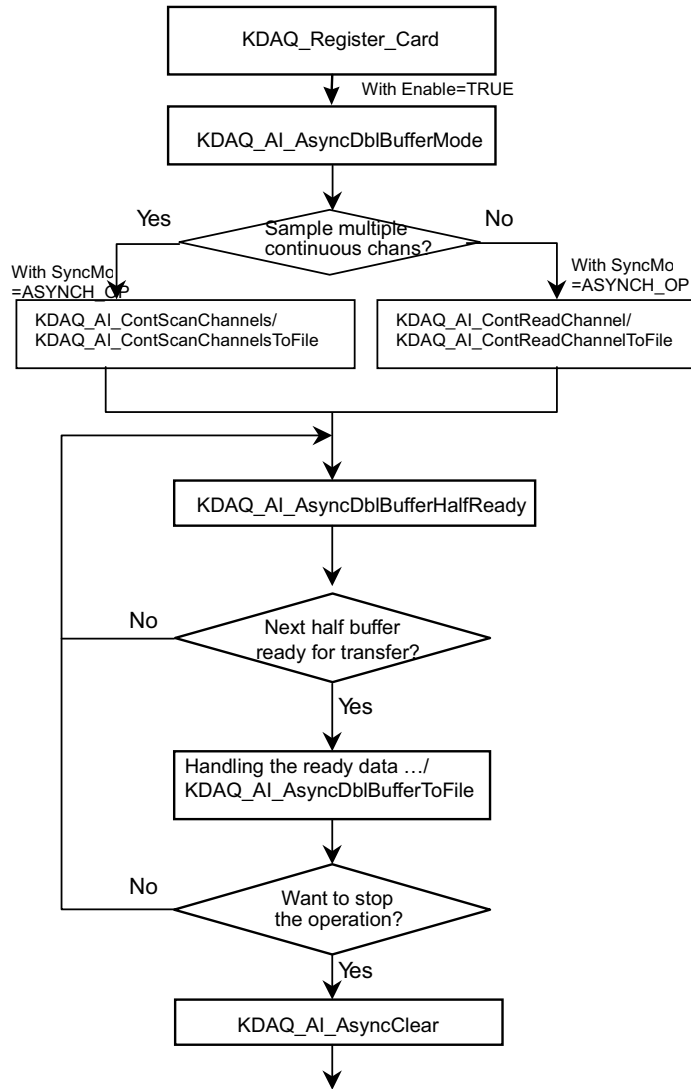
**Example code fragment**

```

card = KDAQ_Register_Card(KPXI_SDAQ_4_2M, card_number);
KDAQ_AI_ContBufferSetup (card, ai_buf, data_size, &BufId);
KDAQ_AI_ContScanChannels (card, channel, BufId, data_size/(channel+1),
ScanIntrv, SampIntrv, ASYNCH_OP); or
KDAQ_AI_ContReadChannel(card, channel, BufId, data_size, ScanIntrv, SampIntrv,
ASYNCH_OP)
do {
    KDAQ_AI_AsyncCheck(card, &bStopped, &count);
} while (!bStopped);

KDAQ_AI_AsyncClear(card, &StartPos, &count);
...
KDAQ_Release_Card(card);
    
```

Figure A-8  
**Double buffered asynchronous operation**



**Example code fragment**

```

card = KDAQ_Register_Card(KPXI_SDAQ_4_2M, card_number);
...
KDAQ_AI_AsyncDblBufferMode (card, 1); // Double-buffered AI
KDAQ_AI_ContBufferSetup (card, ai_buf, data_size, &BufId);
KDAQ_A_ContBufferSetup (card, ai_buf2, data_size, &BufId);
KDAQ_AI_ContScanChannels (card, channel, BufId, data_size/(channel+1),
ScanIntrv, SampIntrv, ASYNCH_OP); or
KDAQ_AI_ContReadChannel(card, channel, BufId, data_size, ScanIntrv, SampIntrv,
ASYNCH_OP)
do {
    do {
        KDAQ_AI_AsyncDblBufferHalfReady(card, &HalfReady, &fstop);
    } while (!HalfReady);

    //Handling the ready data
    ...
} while (!clear_op);

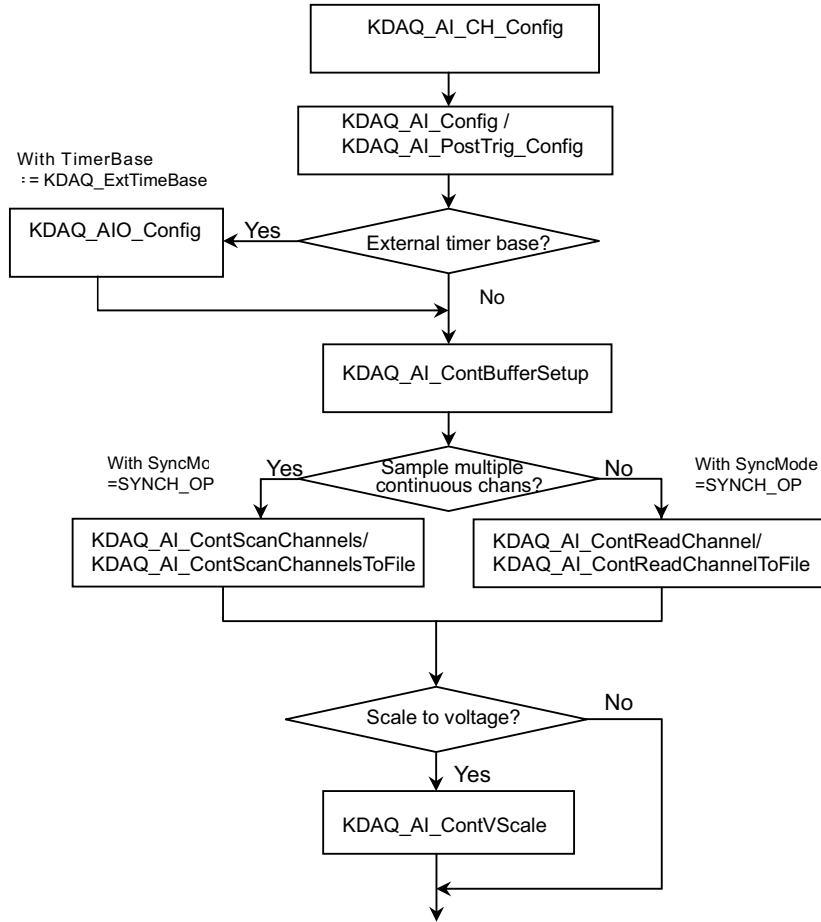
KDAQ_AI_AsyncClear(card, &startPos, &count);
...
KDAQ_Release_Card(card);

```

**Post trigger mode/ delay trigger mode synchronous continuous analog input programming scheme**

This programming section describes the typical flow of post trigger or delay triggered synchronous analog input operation. While performing continuous AI operation, the AI configuration function has to be called at the beginning of the application. In addition, for synchronous AI, the **SyncMode** argument in continuous AI functions has to be set as SYNCH\_OP.

Figure A-9  
**All types of KPXI-DRVR series**





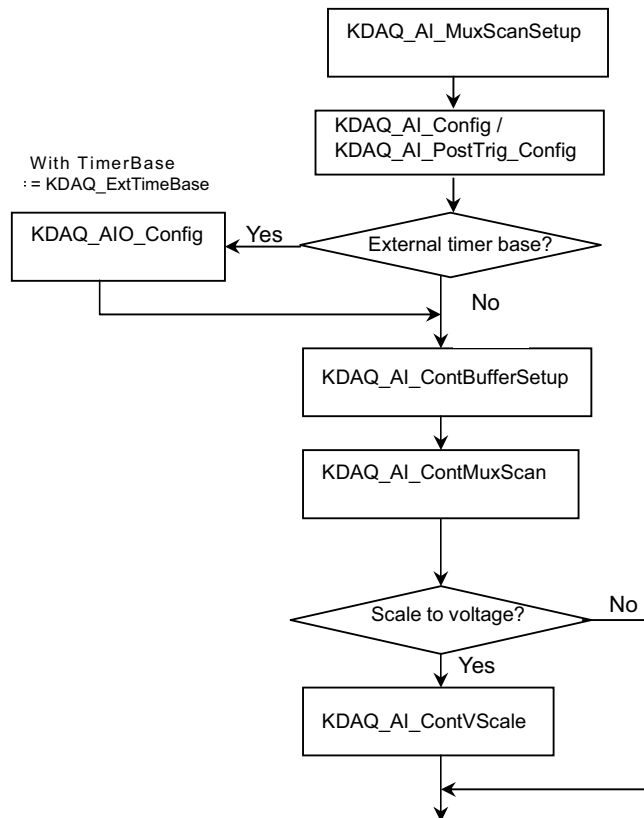
**Example code fragment**

```

card = KDAQ_Register_Card(KPXI_SDAQ_4_2M, card_number);
KDAQ_AI_CH_Config (card, channel, range )
KDAQ_AI_Config (card, 0, KDAQ_AI_TRGMOD_POST| KDAQ_AI_TRGSRC_ExtD|
KDAQ_AI_TrgPositive, 0, 0, 0, 1);
// or
// KDAQ_AI_PostTrig_Config (card, KDAQ_AI_ADCONVSRC_Int, KDAQ_AI_TRGSRC_ExtD|
KDAQ_AI_TrgPositive, 0, 0, 1);
KDAQ_AI_ContBufferSetup (card, ai_buf, data_size, &Id);
KDAQ_AI_ContScanChannels (card, channel, Id, data_size/(channel+1), scan_intrv,
samp_intrv, SYNCH_OP); or
KDAQ_AI_ContReadChannel(card, channel, Id, data_size, scan_intrv, samp_intrv,
SYNCH_OP)
...
KDAQ_Release_Card(card);
    
```

Figure A-10  
**Fills channel gain queue first**

**NOTE:** Only the following models have the **Fills channel gain queue first** feature:  
 KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M.



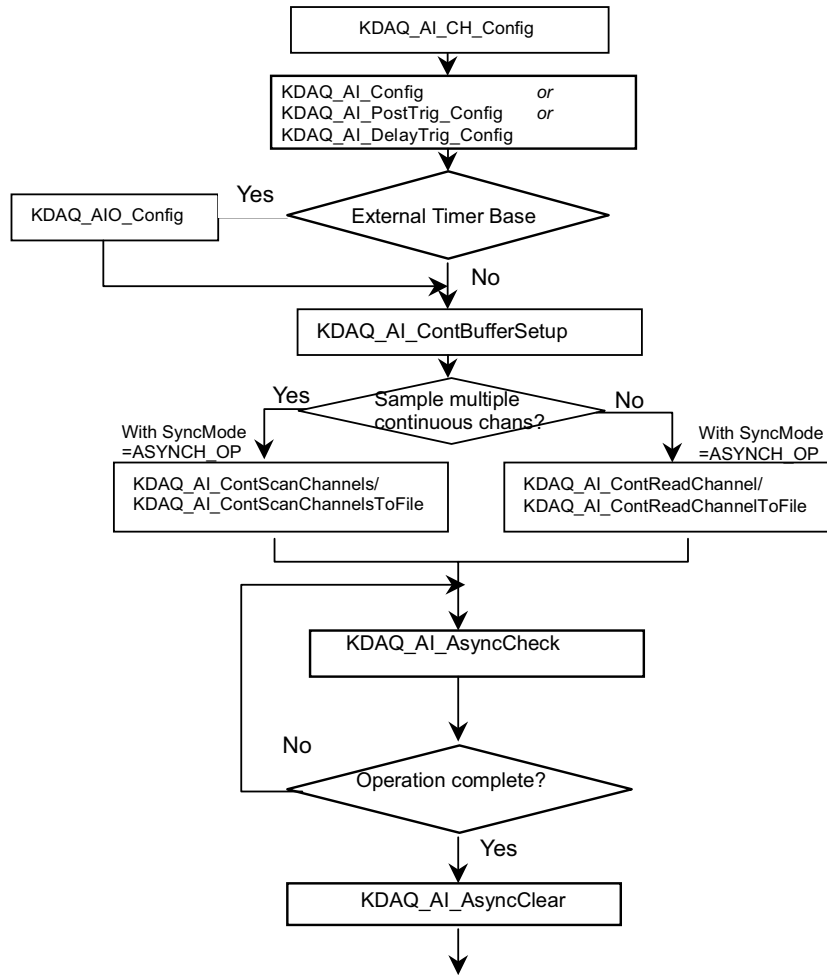
### Example code fragment

```
card = KDAQ_Register_Card(KPXI_DAQ_64_500K, card_number);
CHANNELCOUNT = 1;
chans[0] = 0;
ranges[0] = AD_B_10_V| AI_RSE;
KDAQ_AI_MuxScanSetup(card, CHANNELCOUNT, chans, ranges);
KDAQ_AI_Config (card, KDAQ_AI_ADCONVSRC_Int, KDAQ_AI_TRGMOD_POST|
KDAQ_AI_TRGSRC_ExtD| KDAQ_AI_TrgPositive, 0, 0, 0, 1);
// or
// KDAQ_AI_PostTrig_Config (card, KDAQ_AI_ADCONVSRC_Int, KDAQ_AI_TRGSRC_ExtD|
KDAQ_AI_TrgPositive, 0, 0, 1);
KDAQ_AI_ContBufferSetup (card, ai_buf, data_size, &Id);
KDAQ_AI_ContMuxScan (card, Id, data_size/CHANNELCOUNT,
SAMPLE_INTERVAL*CHANNELCOUNT, SAMPLE_INTERVAL, SYNCH_OP);
...
KDAQ_Release_Card(card);
```

### Post trigger mode/ delay trigger mode non-double-buffered asynchronous continuous analog input programming scheme

This programming section describes the typical flow of post trigger or delay triggered, non-double-buffered asynchronous analog input operation. While performing continuous AI operation, the AI configuration function has to be called at the beginning of your application. In addition, for asynchronous AI, the **SyncMode** argument in continuous AI functions has to be set as **ASYNCH\_OP**.

Figure A-11  
All types of KDAQ-DRVR series



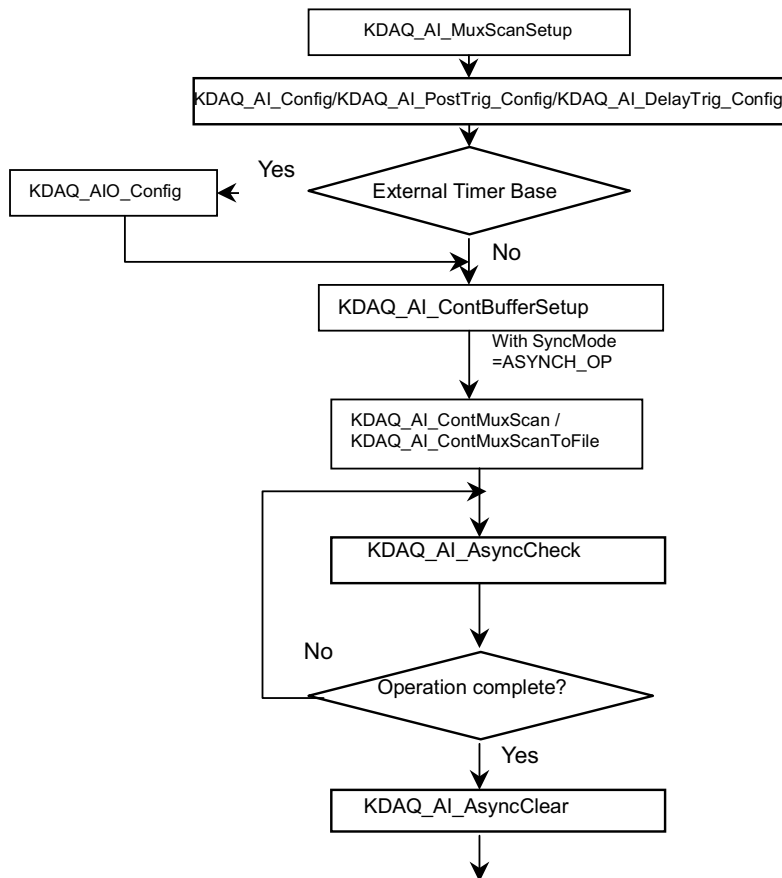
**Example code fragment**

```

card = KDAQ_Register_Card(KPXI_SDAQ_4_2M, card_number);
...
KDAQ_AI_CH_Config (card, channel, range );
KDAQ_AI_Config (card, 0, KDAQ_AI_TRGMOD_POST| KDAQ_AI_TRGSRC_ExtD|
KDAQ_AI_TrigPositive, 0, 0, 0, 1);
// or
// KDAQ_AI_PostTrig_Config (card, KDAQ_AI_ADCONVSRC_Int, KDAQ_AI_TRGSRC_ExtD|
KDAQ_AI_TrigPositive, 0, 0, 1);
KDAQ_AI_AsyncDblBufferMode (card, 0); //non-double-buffered AI
KDAQ_AI_ContBufferSetup (card, ai_buf, data_size, &BufId);
KDAQ_AI_ContScanChannels (card, channel, BufId, data_size/(channel+1),
ScanIntrv, SampIntrv, ASYNCH_OP); or
KDAQ_AI_ContReadChannel(card, channel, BufId, data_size, ScanIntrv, SampIntrv,
ASYNCH_OP);
do {
    KDAQ_AI_AsyncCheck(card, &bStopped, &count);
} while (!bStopped);
KDAQ_AI_AsyncClear(card, &StartPos, &count);
...
KDAQ_Release_Card(card);
  
```

Figure A-12  
**Fills channel gain queue first**

**NOTE:** Only the following models have the **Fills channel gain queue first** feature:  
 KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M.



### Example code fragment

```

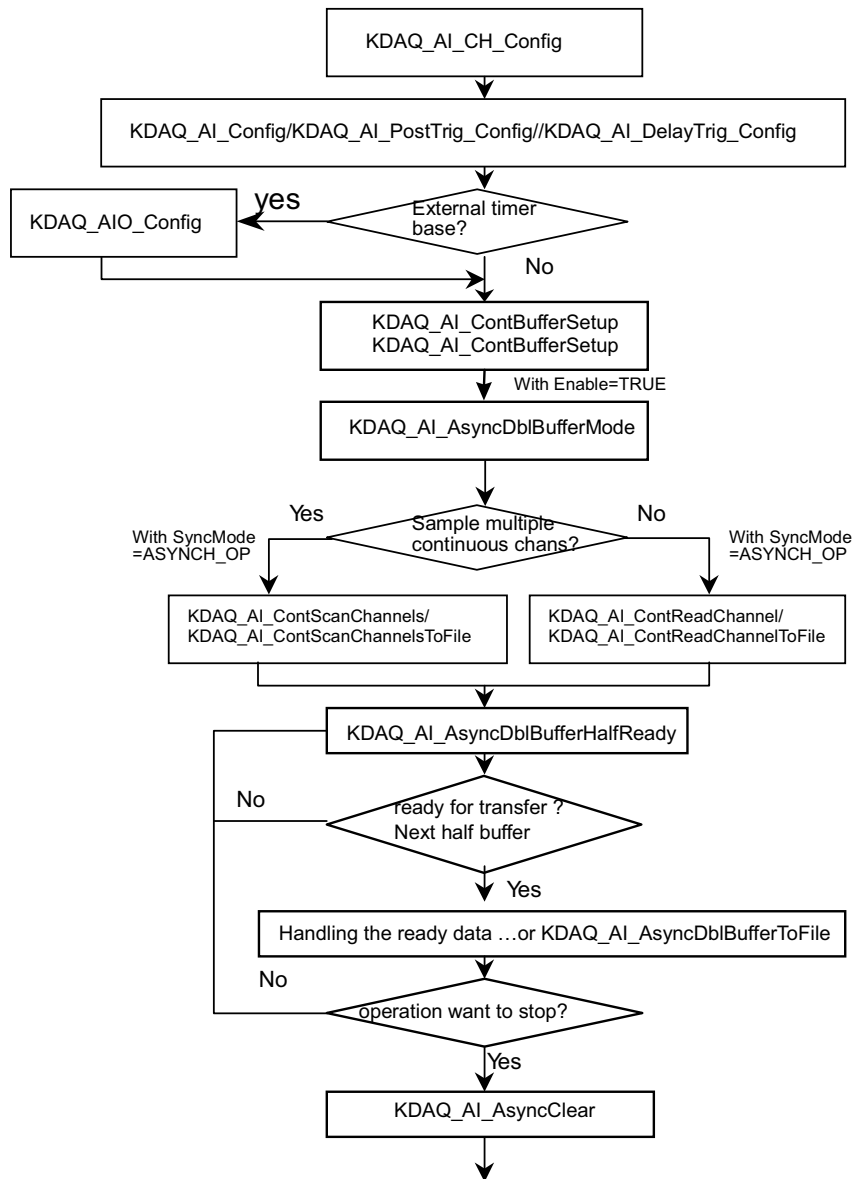
card = KDAQ_Register_Card(KPXI_DAQ_64_500K, card_number);
...
CHANNELCOUNT = 1;
chans[0] = 0;
ranges[0] = AD_B_10_V| AI_RSE;
KDAQ_AI_MuxScanSetup(card, CHANNELCOUNT, chans, ranges);
KDAQ_AI_Config (card, 0, KDAQ_AI_TRGMOD_POST| KDAQ_AI_TRGSRC_ExtD|
KDAQ_AI_TrgPositive, 0, 0, 0, 1);
// or
// KDAQ_AI_PostTrig_Config (card, KDAQ_AI_ADCONVSRC_Int, KDAQ_AI_TRGSRC_ExtD|
KDAQ_AI_TrgPositive, 0, 0, 1);
KDAQ_AI_AsyncDblBufferMode (card, 0); //non-double-buffered AI
KDAQ_AI_ContBufferSetup (card, ai_buf, data_size, &BufId);
KDAQ_AI_ContMuxScan (card, BufId, data_size/(channel+1), ScanIntrv, SampIntrv,
ASYNCH_OP);
do {
    KDAQ_AI_AsyncCheck(card, &bStopped, &count);
} while (!bStopped);
KDAQ_AI_AsyncClear(card, &StartPos, &count);
...
KDAQ_Release_Card(card);

```

### Post trigger mode/ delay trigger mode double-buffered asynchronous continuous analog input programming scheme

This section describes the typical flow of post trigger or delay triggered, double-buffered asynchronous analog input operation. While performing continuous AI operation, the AI configuration function has to be called at the beginning of the application. For asynchronous AI, the **SyncMode** argument in continuous AI functions has to be set as **ASYNCH\_OP**. In addition, double-buffered AI operation is enabled by setting the **KDAQ\_AI\_AsyncDblBufferMode** enable argument to 1.

Figure A-13  
All types of KDAQ-DRVR series



**Example code fragment**

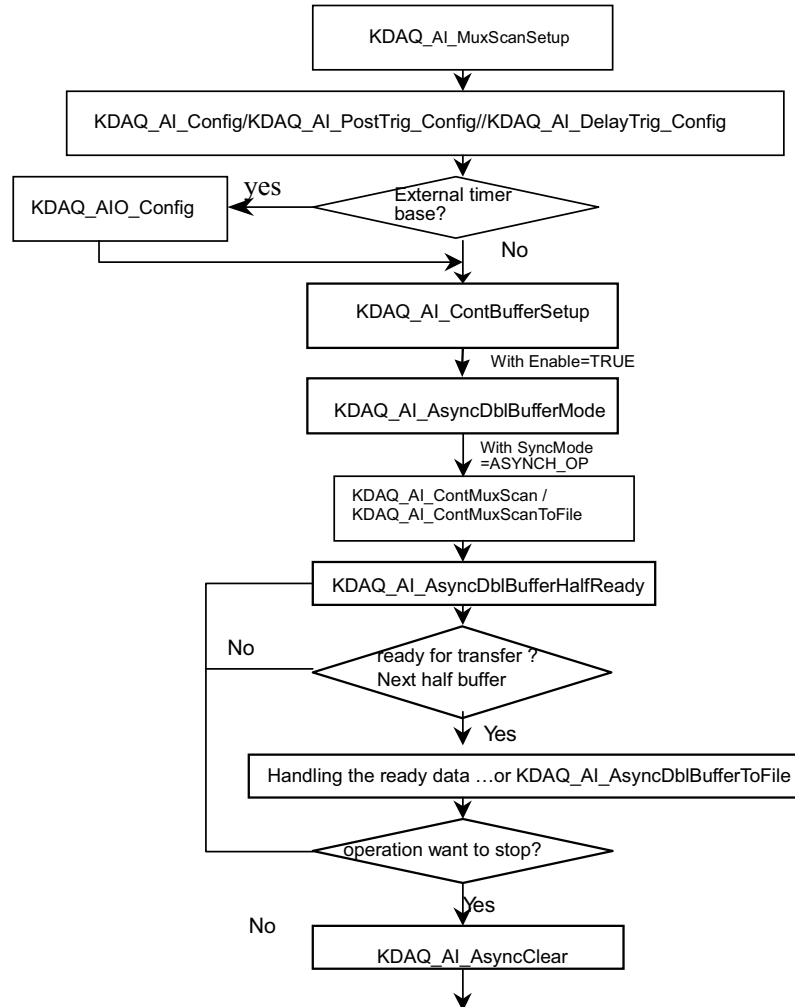
```
card = KDAQ_Register_Card(KPXI_SDAQ_4_2M , card_number);
...1
KDAQ_AI_CH_Config (card, channel, range);
KDAQ_AI_Config (card, 0, KDAQ_AI_TRGMOD_POST| KDAQ_AI_TRGSRC_ExtD|
KDAQ_AI_TrgPositive, 0, 0, 0, 1);
// or
// KDAQ_AI_PostTrig_Config (card, KDAQ_AI_ADCONVSRC_Int, KDAQ_AI_TRGSRC_ExtD|
KDAQ_AI_TrgPositive, 0, 0, 1);
KDAQ_AI_AsyncDblBufferMode (card, 1); // Double-buffered AI
KDAQ_AI_ContBufferSetup (card, ai_buf, data_size, &BufId);
KDAQ_A_ContBufferSetup (card, ai_buf2, data_size, &BufId);
KDAQ_AI_ContScanChannels (card, channel, BufId, data_size/(channel+1),
ScanIntrv, SampIntrv, ASYNCH_OP); or
KDAQ_AI_ContReadChannel(card, channel, BufId, data_size, ScanIntrv, SampIntrv,
ASYNCH_OP);
do {
    do {
        KDAQ_AI_AsyncDblBufferHalfReady(card, &HalfReady, &fstop);
    } while (!HalfReady);

    //Handling the ready data
    ...
} while (!clear_op);

KDAQ_AI_AsyncClear(card, &startPos, &count);
...
KDAQ_Release_Card(card);
```

Figure A-14  
**Fills channel gain queue first**

**NOTE:** Only the following models have the **Fills channel gain queue first** feature: KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M.



**Example code fragment**

```

card = KDAQ_Register_Card(KPXI_DAO_64_500K, card_number);
...
CHANNELCOUNT = 1;
chans[0] = 0;
ranges[0] = AD_B_10_V| AI_RSE;
KDAQ_AI_MuxScanSetup(card, CHANNELCOUNT, chans, ranges);
KDAQ_AI_Config (card, 0, KDAQ_AI_TRGMOD_POST| KDAQ_AI_TRGSRC_ExtD|
KDAQ_AI_TrgPositive, 0, 0, 0, 1);
// or
// KDAQ_AI_PostTrig_Config (card, KDAQ_AI_ADCONVSRC_Int, KDAQ_AI_TRGSRC_ExtD|
KDAQ_AI_TrgPositive, 0, 0, 1);
KDAQ_AI_AsyncDblBufferMode (card, 1); // Double-buffered AI
KDAQ_AI_ContBufferSetup (card, ai_buf, data_size, &BufId);
KDAQ_A_ContBufferSetup (card, ai_buf2, data_size, &BufId);
KDAQ_AI_ContMuxScan (card, BufId, data_size/(channel+1), ScanIntrv, SampIntrv,
ASYNCH_OP);
do {
    do {
        KDAQ_AI_AsyncDblBufferHalfReady(card, &HalfReady, &fstop);
    } while (!HalfReady);

    //Handling the ready data
    ...
} while (!clear_op);

KDAQ_AI_AsyncClear(card, &startPos, &count);
...
KDAQ_Release_Card(card);

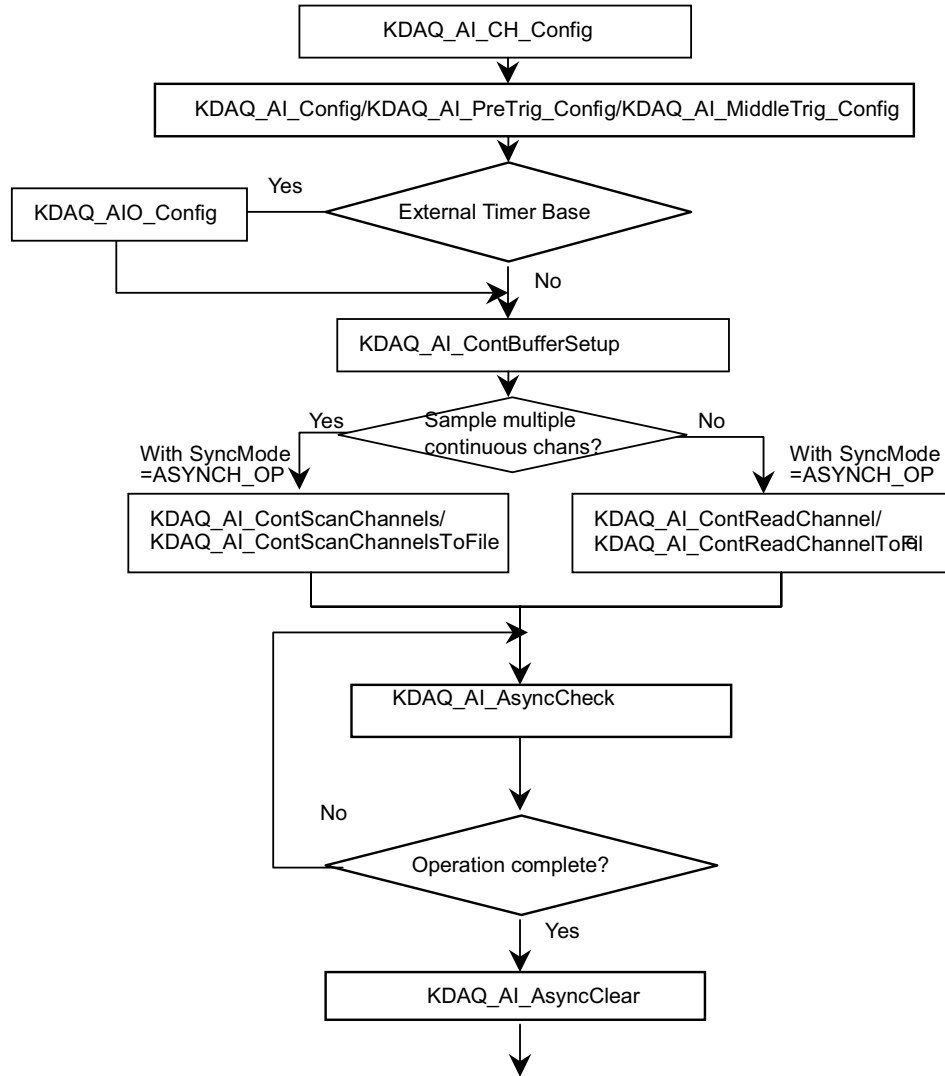
```

**Pre-trigger mode/ middle-trigger mode non-double-buffered asynchronous continuous analog input programming scheme**

This programming section describes the typical flow of pre-trigger and middle trigger mode double-buffered asynchronous analog input operation. A trigger is an event that occurs based on a specified set of conditions. An interrupt mode or DMA-mode Analog input operation can use a trigger to determine when acquisition stop. The trigger mode data acquisition programming is almost the same as the non-trigger mode asynchronous analog input programming. Using KDAQ-DRVR to perform pre-trigger or middle mode data acquisition, the **SyncMode** of continuous AI should be set as ASYNCH\_OP.



Figure A-15  
All types of KDAQ-DRVR series



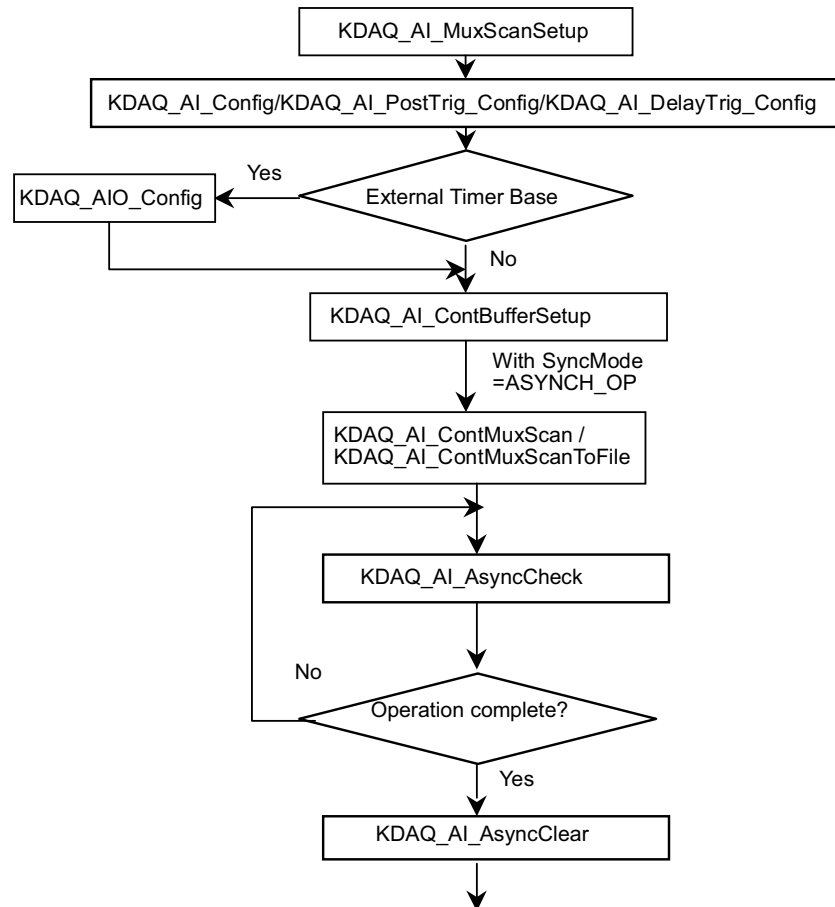
**Example code fragment**

```

card = KDAQ_Register_Card(KPXI_SDAQ_4_2M, card_number);
...
KDAQ_AI_CH_Config (card, channel, range )
KDAQ_AI_Config (card, 0, KDAQ_AI_TRGMOD_PRE| KDAQ_AI_TRGSRC_ExtD|
KDAQ_AI_TrgPositive, 0, 0, 0, 1);
// or
// KDAQ_AI_PreTrig_Config (card, KDAQ_AI_ADCONVSRC_Int, KDAQ_AI_TRGSRC_ExtD|
KDAQ_AI_TrgPositive, 0, 0, 1);
KDAQ_AI_AsyncDblBufferMode (card, 0); //non-double-buffered AI
KDAQ_AI_ContBufferSetup (card, ai_buf, data_size, &BufId);
KDAQ_AI_ContScanChannels (card, channel, BufId, data_size/(channel+1),
ScanIntrv, SampIntrv, ASYNCH_OP); or
KDAQ_AI_ContReadChannel(card, channel, BufId, data_size, ScanIntrv, SampIntrv,
ASYNCH_OP)
do {
    KDAQ_AI_AsyncCheck(card, &bStopped, &count);
} while (!bStopped);
KDAQ_AI_AsyncClear(card, &startPos, &count);
...
KDAQ_Release_Card(card);
    
```

Figure A-16  
**Fills channel gain queue first**

**NOTE:** Only the following models have the **Fills channel gain queue first** feature: KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M



### Example code fragment

```

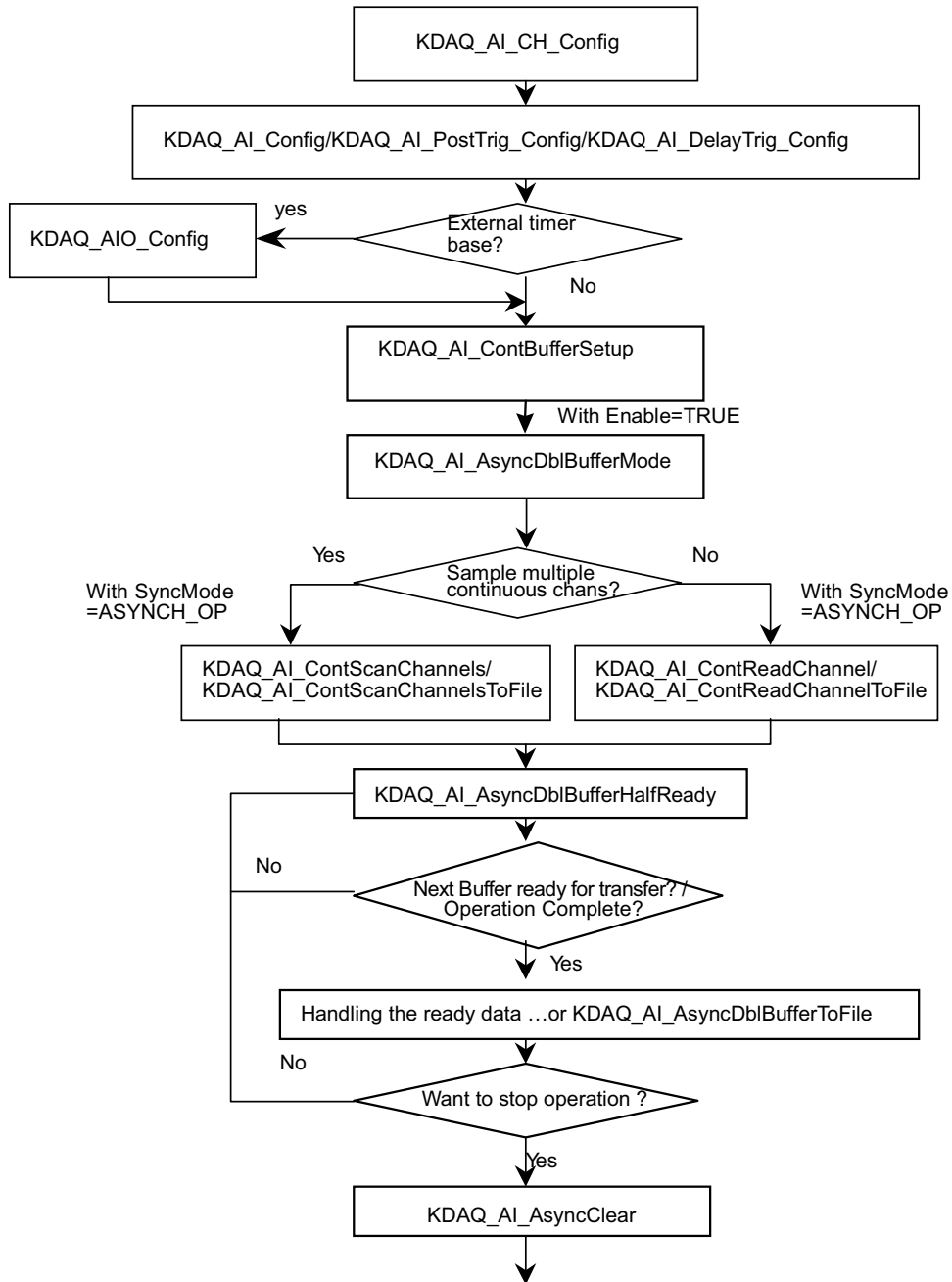
card = KDAQ_Register_Card(KPXI_DAQ_64_500K, card_number);
...
CHANNELCOUNT = 1;
chans[0] = 0;
ranges[0] = AD_B_10_V| AI_RSE;
KDAQ_AI_MuxScanSetup(card, CHANNELCOUNT, chans, ranges);
KDAQ_AI_Config (card, 0, KDAQ_AI_TRGMOD_MIDL| KDAQ_AI_TRGSRC_ExtD|
KDAQ_AI_TrgPositive, POSTCOUNT, 0, 0, 1);
// or
// KDAQ_AI_MiddleTrig_Config (card, KDAQ_AI_ADCONVSRC_Int, KDAQ_AI_TRGSRC_ExtD|
KDAQ_AI_TrgPositive, POSTCOUNT, 0, 0, 1);
KDAQ_AI_AsyncDblBufferMode (card, 0); //non-double-buffered AI
KDAQ_AI_ContBufferSetup (card, ai_buf, data_size, &BufId);
KDAQ_AI_ContMuxScan (card, BufId, data_size/(channel+1), ScanIntrv, SampIntrv,
ASYNCH_OP);
do {
    KDAQ_AI_AsyncCheck(card, &bStopped, &count);
} while (!bStopped);

KDAQ_AI_AsyncClear(card, &startPos, &count);
...
KDAQ_Release_Card(card);
  
```

### Pre-trigger mode/ middle-trigger mode double-buffered asynchronous continuous analog input programming scheme

This programming section describes the typical flow of trigger mode double-buffered asynchronous analog input operation. A trigger is an event that occurs based on a specified set of conditions. An interrupt mode or DMA-mode Analog input operation can use a trigger to determine when acquisition stops. The trigger mode data acquisition programming is almost the same as the non-trigger mode asynchronous analog input programming. Using KDAQ-DRVR to perform trigger mode data acquisition, the **SyncMode** of continuous AI should be set as ASYNCH\_OP. In addition, double-buffered AI operation is enabled by setting enable argument of **KDAQ\_AI\_AsyncDbIBufferMode** function to 1.

Figure A-17  
All types of KDAQ-DRVR series



**Example code fragment**

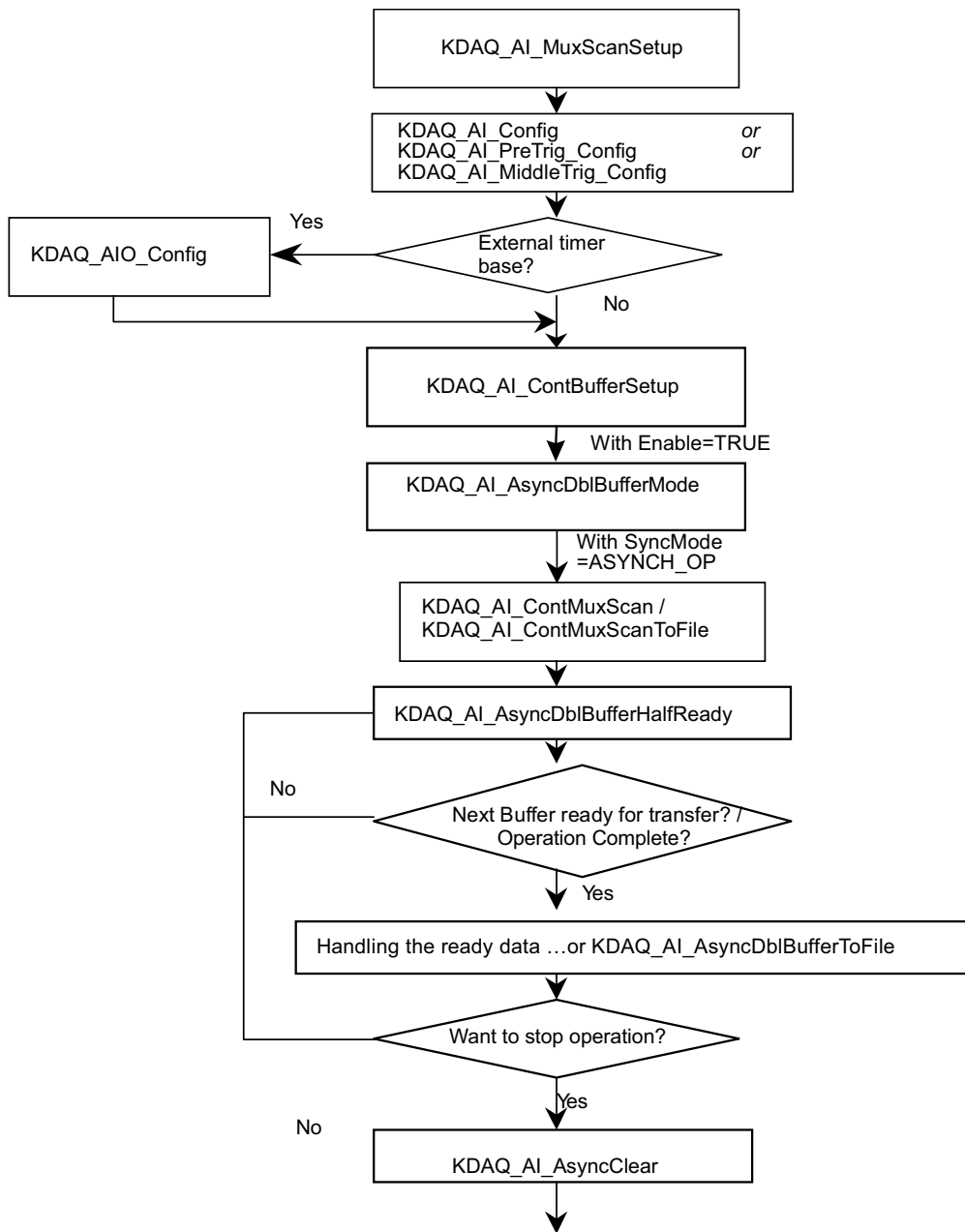
```
card = KDAQ_Register_Card(KPXI_SDAQ_4_2M, card_number);
...
KDAQ_AI_CH_Config (card, channel, range )
KDAQ_AI_Config (card, 0, KDAQ_AI_TRGMOD_PRE|KDAQ_AI_TRGSRC_ExtD, 0, 0, 0, 1);
// or
// KDAQ_AI_MiddleTrig_Config (card, KDAQ_AI_ADCONVSRC_Int, KDAQ_AI_TRGSRC_ExtD|
KDAQ_AI_TrgPositive, 0, 0, 1);
AI_AsyncDblBufferMode (card, 1); Double-buffered AI
KDAQ_AI_ContBufferSetup (card, ai_buf, data_size, &BufId);
KDAQ_A_ContBufferSetup (card, ai_buf2, data_size, &BufId);
KDAQ_AI_ContScanChannels (card, channel, BufId, data_size/(channel+1),
ScanIntrv, U32 SampIntrv, ASYNCH_OP); or
KDAQ_AI_ContReadChannel (card, channel, BufId, data_size, ScanIntrv, U32
SampIntrv, ASYNCH_OP)
do {
    do {
        KDAQ_AI_AsyncDblBufferHalfReady (card, &HalfReady, &fstop);
    } while (!HalfReady && !fstop);

    //handling the ready data ...
    ...
} while (!clear_op && !fstop);

KDAQ_AI_AsyncClear (card, &startPos, &count);
...
KDAQ_Release_Card (card);
```

Figure A-18  
**Fills channel gain queue first**

**NOTE:** Only the following models have the **Fills channel gain queue first** feature: KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M.



**Example code fragment**

```

card = KDAQ_Register_Card(KPXI_DAQ_64_500K, card_number);
...
CHANNELCOUNT = 1;
chans[0] = 0;
ranges[0] = AD_B_10_V| AI_RSE;
KDAQ_AI_MuxScanSetup(card, CHANNELCOUNT, chans, ranges);
KDAQ_AI_Config (card, 0, KDAQ_AI_TRGMOD_RRE| KDAQ_AI_TRGSRD_ExtD|
KDAQ_AI_TrgPositive, 0, 0, 0, 1);
// or
    
```

```

// KDAQ_AI_PreTrig_Config (card, KDAQ_AI_ADCONVSRV_Int, KDAQ_AI_TRGSRV_ExtD|
KDAQ_AI_TrgPositive, 0, 0, 1);
KDAQ_AI_AsyncDblBufferMode (card, 1); // Double-buffered AI
KDAQ_AI_ContBufferSetup (card, ai_buf, data_size, &BufId);
KDAQ_A_ContBufferSetup (card, ai_buf2, data_size, &BufId);
KDAQ_AI_ContMuxScan (card, BufId, data_size/(channel+1), ScanIntrv, SampIntrv,
ASYNCH_OP);
do {
    do {
        KDAQ_AI_AsyncDblBufferHalfReady(card, &HalfReady, &fstop);
    } while (!HalfReady);

    //Handling the ready data
    ...
} while (!clear_op);

KDAQ_AI_AsyncClear(card, &startPos, &count);
...
KDAQ_Release_Card(card);

```

## Analog output programming hints

KDAQ-DRVR provides two kinds of analog output operation — non-buffered single-point analog output operation and buffered continuous analog output operation.

The non-buffered single-point AO uses a software polling method to write data to the device. The programming scheme for this kind of AO operation is described in the paragraph titled [One-shot analog output programming scheme](#).

The buffered continuous AO uses DMA transfer method to transfer data from the user's buffer to the device. The maximum number of count in one transfer depends on the size of initially allocated memory for analog output in the driver. Use the **KDAQ\_AO\_InitialMemoryAllocated** function to get the size of initially allocated memory before starting to perform continuous AO operation.

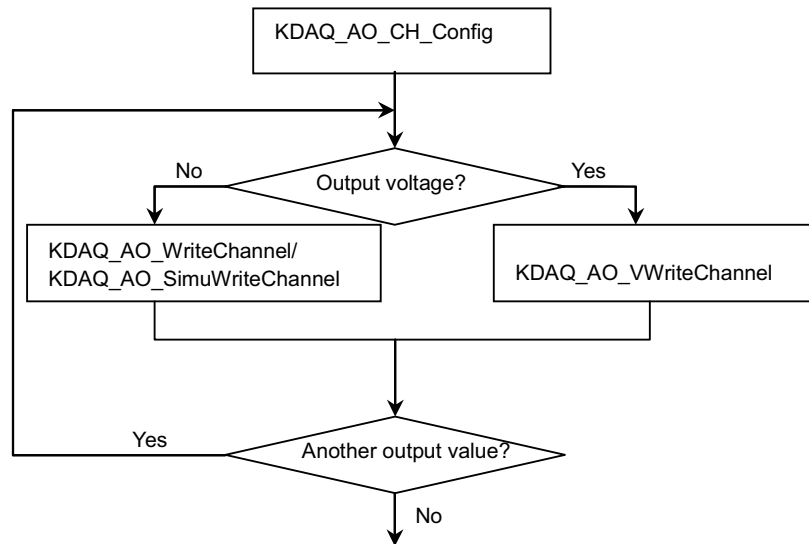
Refer to [Continuous data transfer in KDAQ-DRVR](#) later in this section for special considerations and performance issues for the buffered continuous analog output.

## One-shot analog output programming scheme

The following examples describe the typical flow of non-buffered single-point analog output operations.

Figure A-19  
**One-shot analog output programming**

**NOTE:** *Figure A-19 (and the example code fragment immediately following the figure) details the typical flow of the following models: KPXI-SDAQ-4-500K, KPXI-SDAQ-4-2M, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K*



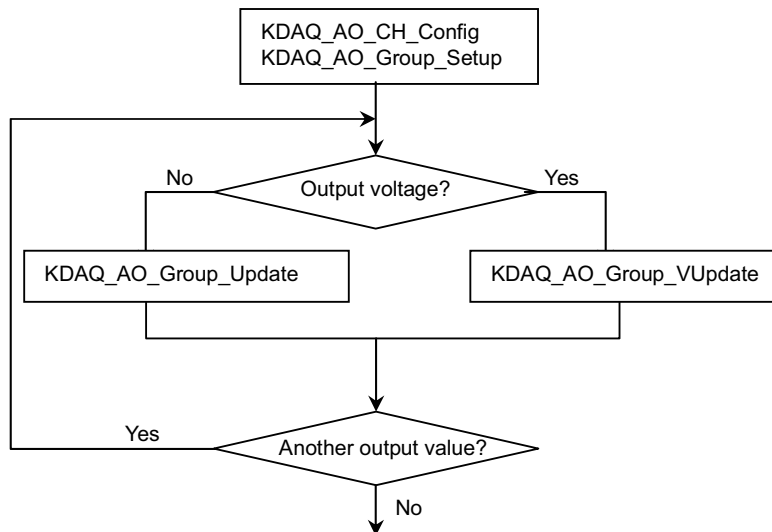
**Example code fragment**

```

card = KDAQ_Register_Card(KPXI_SDAQ_4_2M, card_number);
...
KDAQ_AO_CH_Config (card, 0, KDAQ_DA_BiPolar, KDAQ_DA_Int_REF, 10.0);
KDAQ_AO_WriteChannel(card, chan, out_value);
...
KDAQ_Release_Card(card);
  
```

Figure A-20  
**One-shot analog output programming**

**NOTE:** *Figure A-20 (and the example code fragment immediately following the figure) details the typical flow of the following models: KPXI-AO-4-1M and KPXI-AO-8-1M.*



**Example code fragment**

```

card = KDAQ_Register_Card(KPXI_AO_8_1M, card_number);
    da_ch = 0;
KDAQ_AO_CH_Config (card, da_ch, KDAQ_DA_BiPolar, KDAQ_DA_Int_REF, 10.0);
KDAQ_AO_Group_Setup (card, DA_Group_A, 1, &da_ch);
KDAQ_AO_Group_VUpdate (card, DA_Group_A, &out_V);
...
KDAQ_Release_Card(card);
  
```

**Continuous analog output (with initial default settings) programming scheme**

This programming section describes the typical flow of synchronous analog output operation performed by the device in a default configuration. While performing continuous AO operation, the AO configuration function has to be called at the beginning of the application. In addition, the **SyncMode** argument in continuous AO functions has to be set as **ASYNCH\_OP**.

Table A-3  
**Initial default channel configuration**

Channel	Configuration
D/A Output Polarity	KDAQ_DA_BiPolar
D/A Reference voltage source	KDAQ_DA_Int_REF
D/A Reference voltage value	10.0

Table A-4  
**Initial default DA configuration**

Channel	Configuration
D/A R/W source	KDAQ_DA_WRSRC_Int (Internal timer pacer)
D/A Trigger mode	KDAQ_DA_TRGMOD_POST (post trigger)

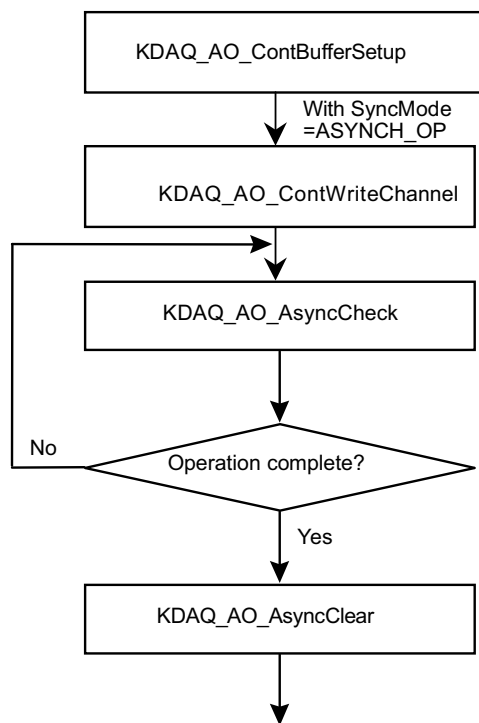


Table A-4 (continued)  
**Initial default DA configuration**

Channel	Configuration
D/A Trigger source	KDAQ_DA_TRGSRC_SOFT (software trigger)
Auto buffer reset	KPXI-AO-4-1M: TRUE KPXI-AO-8-1M: FALSE

**NOTE:** *Figure A-21 (and the example code fragment immediately following the figure) details the typical flow of the following models: KPXI-SDAQ-4-500K, KPXI-SDAQ-4-2M, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K.*

Figure A-21  
**Non-double-buffered asynchronous continuous analog output programming**



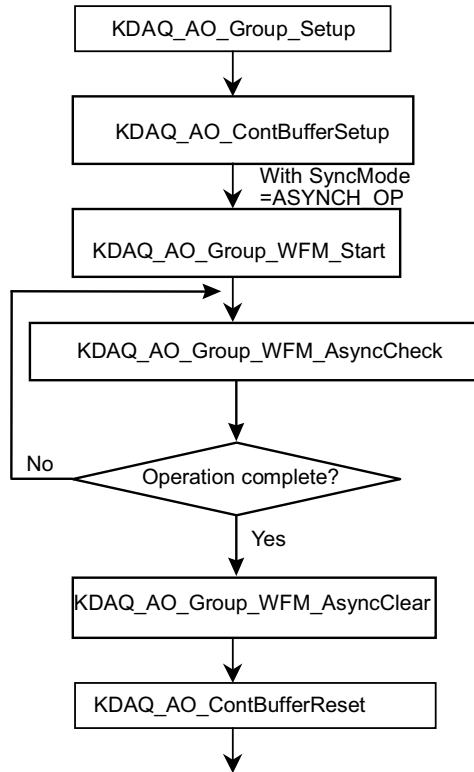
**Example code fragment**

```

card = KDAQ_Register_Card(KPXI_AO_8_1M, card_number);
...
KDAQ_AO_ContBufferSetup (card, ao_buf, data_size, &DaId);
KDAQ_AO_ContWriteChannel(card, 0, DaId, data_size, iteration, samp_intrv,
    samp_intrv, ASYNCH_OP);
do {
    KDAQ_AO_AsyncCheck(card, &bStopped, &count);
} while (!bStopped);
KDAQ_AO_AsyncClear(card, &count, mode);
...
KDAQ_Release_Card(card);
  
```

**NOTE:** *Figure A-22 (and the example code fragment immediately following the figure) details the typical flow of the following models: KPXI-AO-4-1M and KPXI-AO-8-1M.*

Figure A-22  
**Non-double-buffered asynchronous continuous analog output programming**



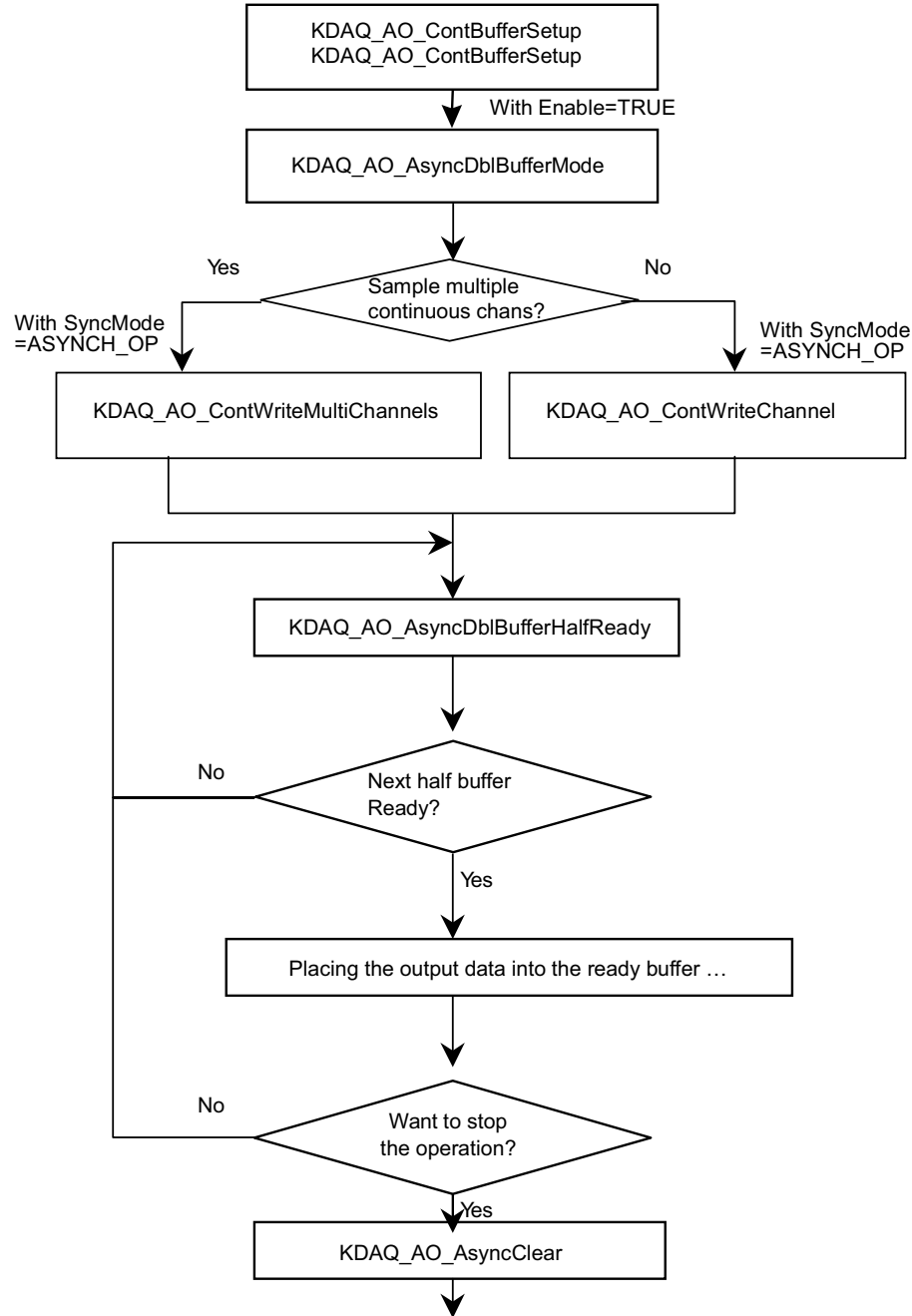
### Example code fragment

```

card = KDAQ_Register_Card(KPXI_AO_8_1M, card_number);
KDAQ_AO_Group_Setup (card, DA_Group_A, 1, &da_ch); //DA channel 0 in group A
KDAQ_AO_ContBufferSetup (card, ao_buf, data_size, &DaId);
KDAQ_AO_Group_WFM_Start (card, DA_Group_A, Id, DaId, data_size/2, 10,
samp_intrv, 1);
do {
    KDAQ_AO_Group_WFM_AsyncCheck(card, DA_Group_A, &bStopped, &count);
    } while (!bStopped);
KDAQ_AO_Group_WFM_AsyncClear(card, DA_Group_A, &count, 0);
KDAQ_AO_ContBufferReset (card);
KDAQ_Release_Card(card);
  
```

**NOTE:** *Figure A-23 (and the example code fragment immediately following the figure) details the typical flow of the following models: KPXI-SDAQ-4-500K, KPXI-SDAQ-4-2M, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K.*

Figure A-23  
**Double-buffered asynchronous continuous analog output programming**



**Example code fragment**

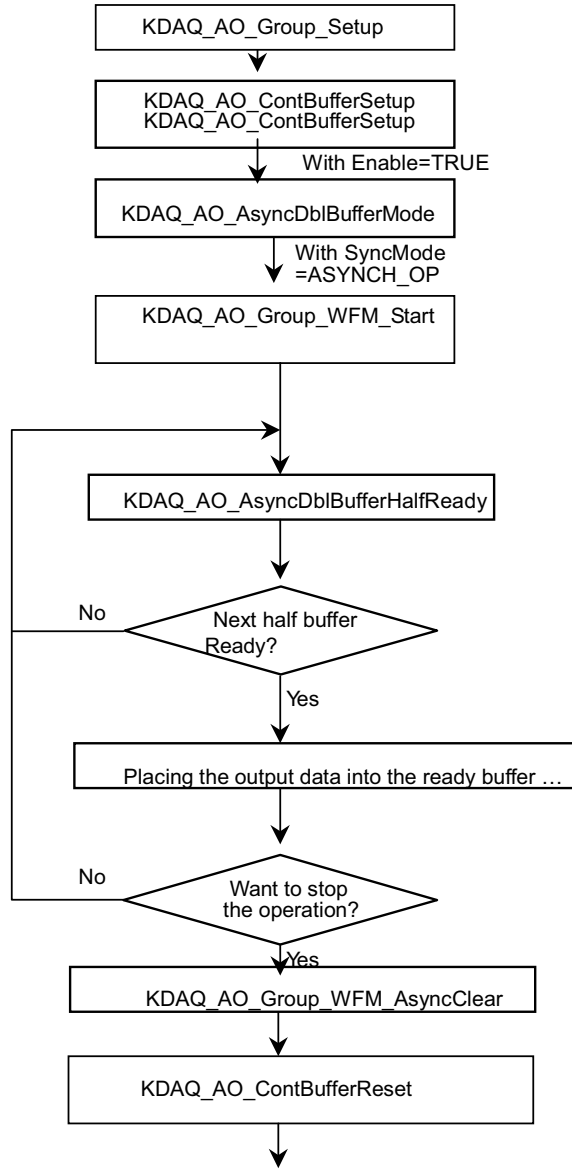
```
card = KDAQ_Register_Card(KPXI_SDAQ_4_2M, card_number);
KDAQ_AO_ContBufferSetup (card, ao_buf, data_size, &DaId);
KDAQ_AO_ContBufferSetup (card, ao_buf2, data_size, &DaId);
KDAQ_AO_AsyncDblBufferMode (card, 1);
KDAQ_AO_ContWriteChannel(card, 0, DaId, data_size, 0, samp_intrv, samp_intrv,
    ASYNCH_OP);
do {
    do {
        KDAQ_AO_AsyncDblBufferHalfReady(card, &HalfReady);
    } while (!HalfReady);

    // Placing the output data into the ready buffer ...
    ...
} while (!clear_op);

KDAQ_AO_AsyncClear(card, &count, mode);
...
KDAQ_Release_Card(card);
```

**NOTE:** [Figure A-24](#) (and the example code fragment immediately following the figure) details the typical flow of the following models: KPXI-AO-4-1M, KPXI-AO-8-1M.

Figure A-24  
**Double-buffered asynchronous continuous analog output programming**



**Example code fragment**

```

card = KDAQ_Register_Card(KPXI_AO_8_1M, card_number);
KDAQ_AO_Group_Setup (card, DA_Group_A, 1, &da_ch); //DA channel 0 in group A
KDAQ_AO_ContBufferSetup (card, ao_buf, data_size, &DaId);
KDAQ_AO_ContBufferSetup (card, ao_buf2, data_size, &DaId);
KDAQ_AO_AsyncDblBufferMode (card, 1);
KDAQ_AO_Group_WFM_Start (card, DA_Group_A, Id, DaId, data_size/2, 0, samp_intrv,
1);
do {
    do {
        KDAQ_AO_AsyncDblBufferHalfReady(card, &HalfReady);
    } while (!HalfReady);

    // Placing the output data into the ready buffer ...
    ...
} while (!clear_op);
    
```

```

KDAQ_AO_Group_WFM_AsyncClear(card, DA_Group_A, &count,0);
  KDAQ_AO_ContBufferReset (card); ...
KDAQ_Release_Card(card);

```

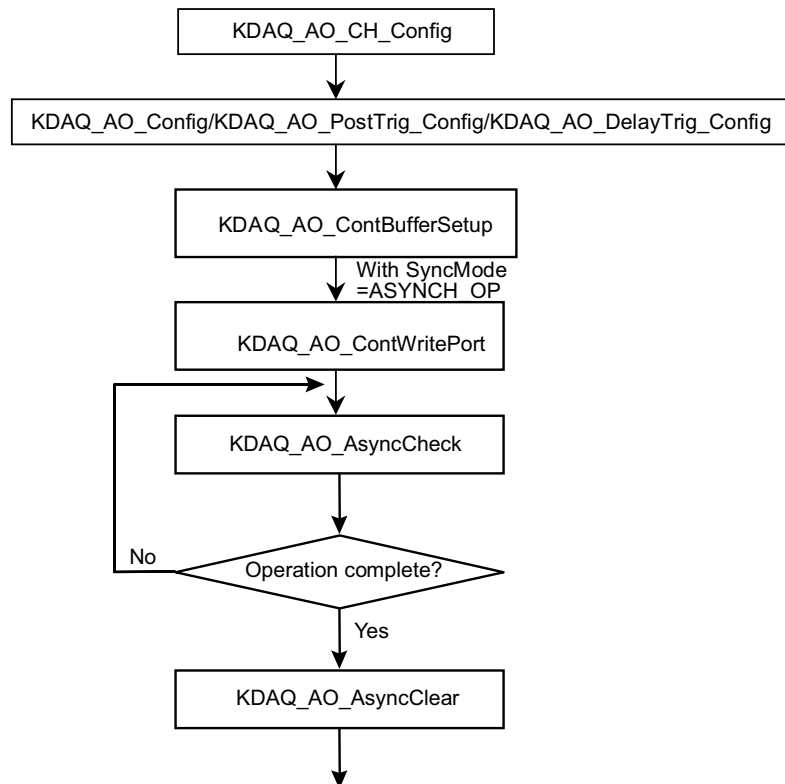
### Non-double-buffered asynchronous continuous analog output programming

This programming section describes the typical flow of asynchronous analog output operation. When performing continuous AO operation, call the AO configuration function at the beginning of your application. In addition, the **SyncMode** argument in continuous AO functions has to be set as **ASYNCH\_OP**.

**NOTE:** *Figure A-25 (and the example code fragment immediately following the figure) details the typical flow of the following models: KPXI-SDAQ-4-500K, KPXI-SDAQ-4-2M, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K.*

Figure A-25

#### Typical flow of asynchronous analog output operation



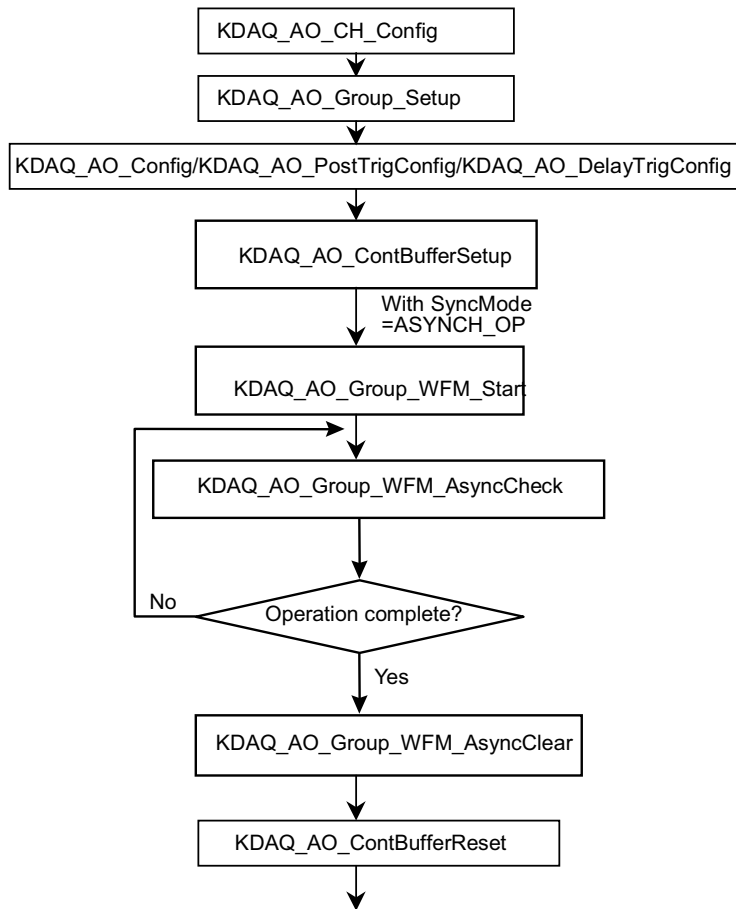
**Example code fragment**

```

card = KDAQ_Register_Card(KPXI_SDAQ_4_2M, card_number);
...
KDAQ_AO_CH_Config (card, 0, KDAQ_DA_BiPolar, KDAQ_DA_Int_REF, 10.0);
KDAQ_AO_Group_Setup (card, DA_Group_A, 1, &da_ch); //DA channel 0 in group A
KDAQ_AO_Config (card, 0, KDAQ_DA_TRGMOD_POST|KDAQ_DA_TRGSRC_ExtD, 1, 0, 0,1)
KDAQ_AO_ContBufferSetup (card, ao_buf, data_size, &DaId);
KDAQ_AO_ContWriteChannel(card, 0, DaId, data_size, iteration, samp_intrv,
    samp_intrv, ASYNCH_OP);
    do {
        KDAQ_AO_AsyncCheck(card, &bStopped, &count);
    } while (!bStopped);
KDAQ_AO_AsyncClear(card, &count, mode);
...
KDAQ_Release_Card(card);
    
```

**NOTE:** *Figure A-26 (and the example code fragment immediately following the figure) details the typical flow of the following models: KPXI-AO-4-1M, KPXI\_AO\_8\_1M.*

Figure A-26  
**Typical flow of asynchronous analog output operation**



### Example code fragment

```
card = KDAQ_Register_Card(KPXI_AO_8_1M, card_number);
...
KDAQ_AO_CH_Config (card, 0, KDAQ_DA_BiPolar, KDAQ_DA_Int_REF, 10.0);
KDAQ_AO_Group_Setup (card, DA_Group_A, 1, &da_ch); //DA channel 0 in group A
KDAQ_AO_Config (card, 0, KDAQ_DA_TRGMOD_POST|KDAQ_DA_TRGSRC_ExtD, 1, 0, 0,0);
KDAQ_AO_ContBufferSetup (card, ao_buf, data_size, &DaId);
KDAQ_AO_Group_WFM_Start (card, DA_Group_A, Id, DaId, data_size/2, 10,
samp_intrv, 1);
    do {
        KDAQ_AO_Group_WFM_AsyncCheck(card, DA_Group_A, &bStopped, &count);
    } while (!bStopped);
KDAQ_AO_Group_WFM_AsyncClear(card, DA_Group_A, &count, 0);
KDAQ_AO_ContBufferReset (card);
KDAQ_Release_Card(card);
```

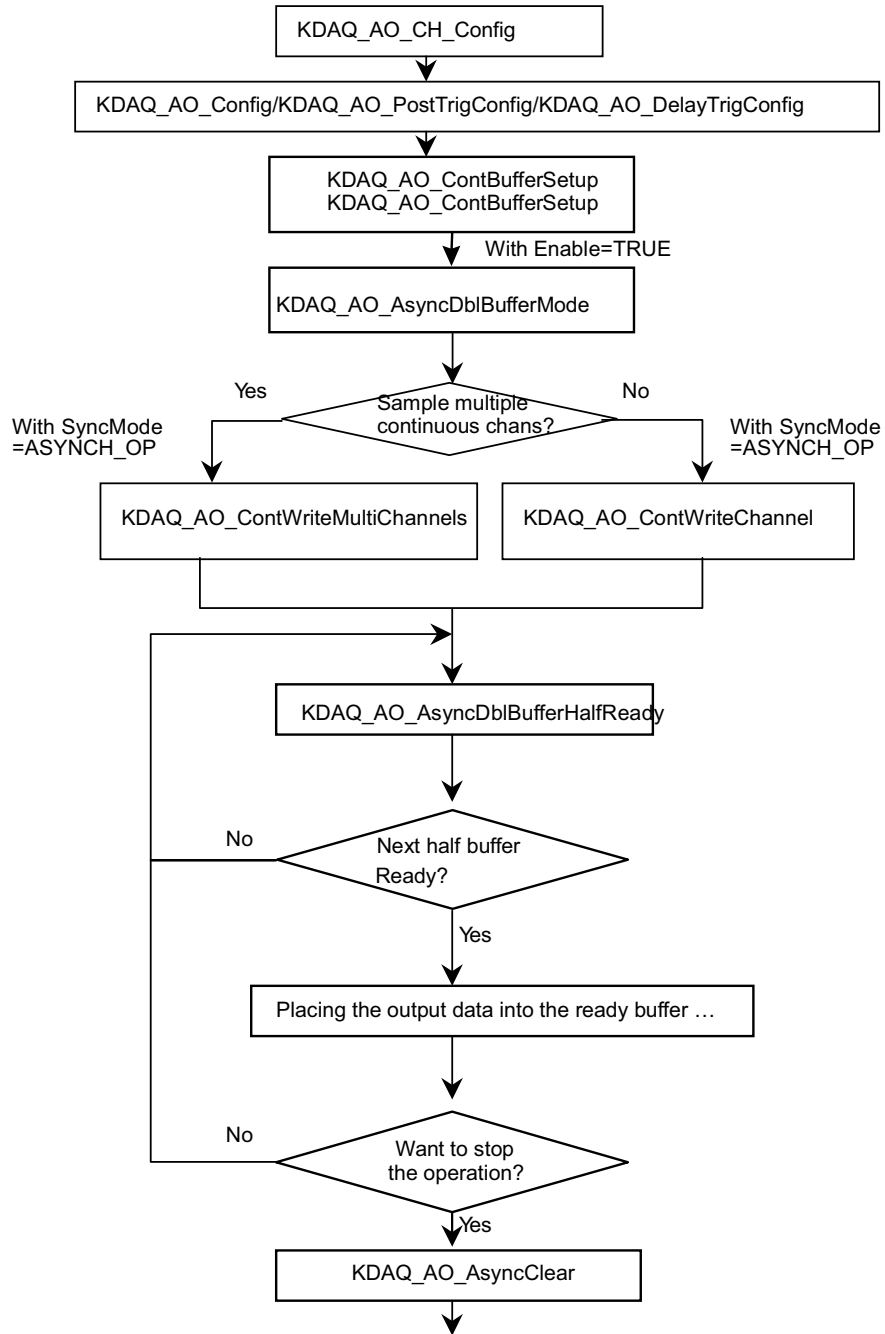
### Double-buffered asynchronous continuous analog output programming

This programming section describes the typical flow of double-buffered asynchronous analog output operation. When performing continuous AO operation, call the AO configuration function at the beginning of your application. The **SyncMode** argument in continuous AO functions has to be set as **ASYNCH\_OP**. In addition, double-buffered AO operation is enabled by setting enable argument of the **KDAQ\_AO\_AsyncDbIBufferMode** function to 1.

**NOTE:** [Figure A-27](#) (and the example code fragment immediately following the figure) details the typical flow of the following models: KPXI-SDAQ-4-500K, KPXI-SDAQ-4-2M, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K.



Figure A-27  
**Typical flow of double-buffered asynchronous analog output operation**



**Example code fragment**

```

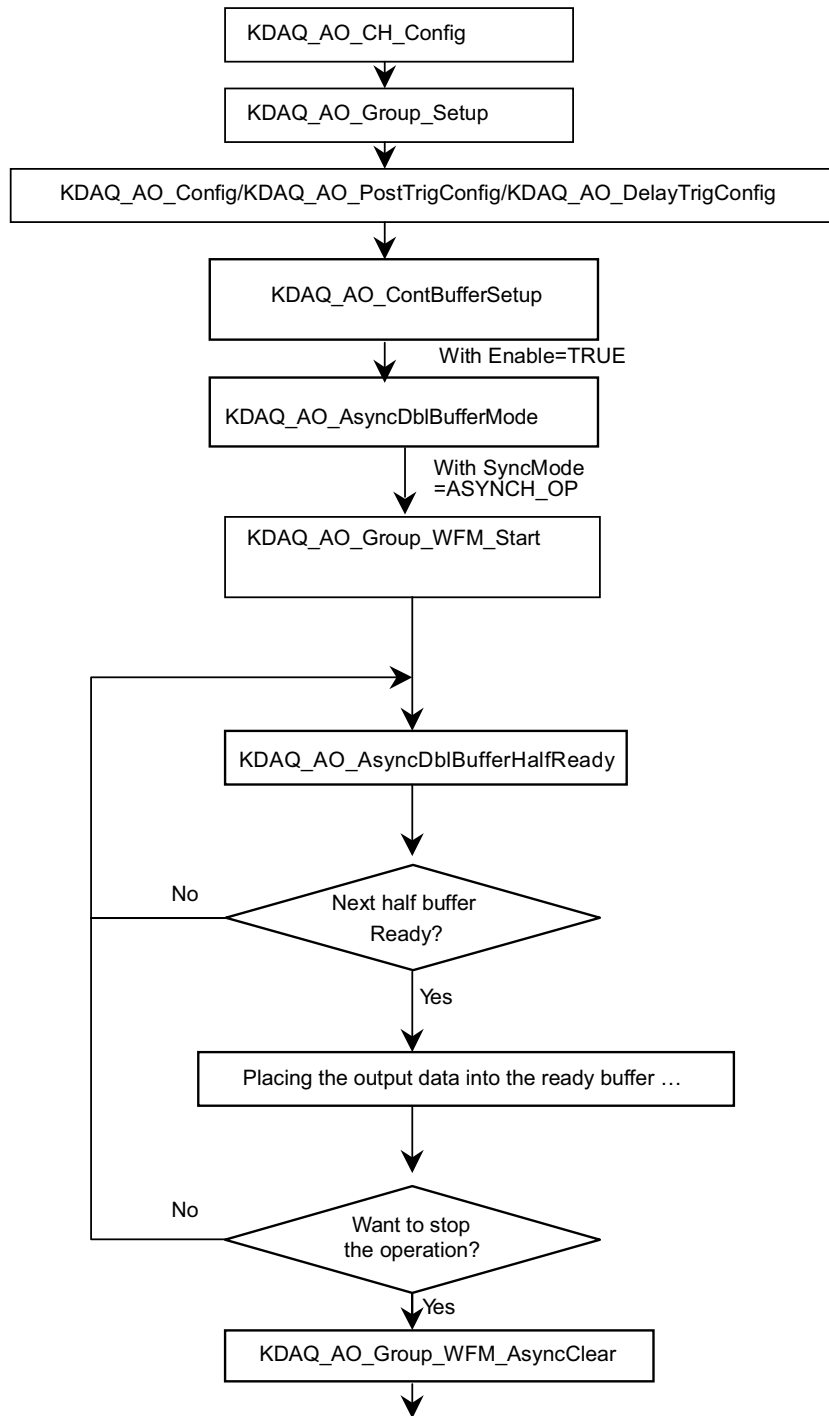
card = KDAQ_Register_Card(KPXI_SDAQ_4_2M, card_number);
...
KDAQ_AO_CH_Config (card, 0, KDAQ_DA_BiPolar, KDAQ_DA_Int_REF, 10.0);
KDAQ_AO_Config (card, 0, KDAQ_DA_TRGMOD_POST|KDAQ_DA_TRGSRC_ExtD, 1, 0, 0,1)
KDAQ_AO_ContBufferSetup (card, ao_buf, data_size, &DaId);
KDAQ_AO_ContBufferSetup (card, ao_buf2, data_size, &DaId);
KDAQ_AO_AsyncDblBufferMode (card, 1);
KDAQ_AO_ContWriteChannel(card, 0, DaId, data_size, 0, samp_intrv, samp_intrv,

```

```
    ASYNCH_OP);
do {
    do {
        KDAQ_AO_AsyncDblBufferHalfReady(card, &HalfReady);
    } while (!HalfReady);
    // Placing the output data into the ready buffer ...
    ...
} while (!clear_op);
KDAQ_AO_AsyncClear(card, &count, mode);
...
KDAQ_Release_Card(card);
```

**NOTE:** [Figure A-28](#) (and the example code fragment immediately following the figure) details the typical flow of the following models: KPXI-AO-4-1M, KPXI-AO-8-1M.

Figure A-28  
Typical flow of double-buffered asynchronous analog output operation



### Example code fragment

```

card = KDAQ_Register_Card(KPXI_AO_8_1M, card_number);
...
KDAQ_AO_CH_Config (card, 0, KDAQ_DA_BiPolar, KDAQ_DA_Int_REF, 10.0);
KDAQ_AO_Group_Setup (card, DA_Group_A, 1, &da_ch); //DA channel 0 in group A
KDAQ_AO_Config (card, 0, KDAQ_DA_TRGMOD_POST|KDAQ_DA_TRGSRC_ExtD, 1, 0, 0,0);
KDAQ_AO_ContBufferSetup (card, ao_buf, data_size, &DaId);
KDAQ_AO_ContBufferSetup (card, ao_buf2, data_size, &DaId);
KDAQ_AO_AsyncDblBufferMode (card, 1);
KDAQ_AO_Group_WFM_Start (card, DA_Group_A, Id, DaId, data_size/2, 0, samp_intrv,
1);
do {
    do {
        KDAQ_AO_AsyncDblBufferHalfReady(card, &HalfReady);
    } while (!HalfReady);

    // Placing the output data into the ready buffer ...
    ...
} while (!clear_op);
KDAQ_AO_Group_WFM_AsyncClear(card, DA_Group_A, &count,0);
    KDAQ_AO_ContBufferReset (card); ...
KDAQ_Release_Card(card);

```

## Digital input programming hints

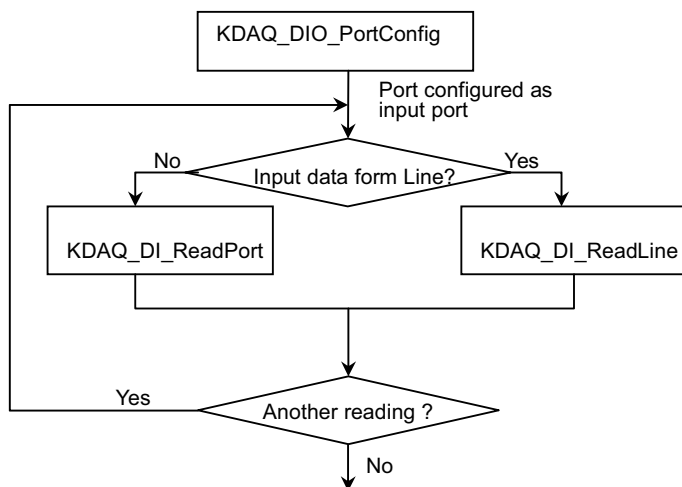
KDAQ-DRVR provides one kind of digital input operation — a non-buffered single-point digital input operation. The non-buffered single-point DI uses a software polling method to read data from the device. The programming scheme for this kind of DI operation is described in [One-shot digital input programming](#).

### One-shot digital input programming

This programming section describes the typical flow of non-buffered single-point digital input readings. When performing one-shot DI operation, for devices whose I/O port can be set as input or output, include port configuration at the beginning of the application. Refer to [Figure A-29](#) and the example code fragment immediately following the figure.

Figure A-29

#### One-shot digital input programming



**Example code fragment**

```

card = KDAQ_Register_Card(KPXI_SDAQ_4_2M , card_number);
//port configured
KDAQ_DIO_PortConfig(card ,Channel_P1A, INPUT_PORT);
KDAQ_DIO_PortConfig(card, Channel_P1B, INPUT_PORT);
KDAQ_DIO_PortConfig(card, Channel_P1CL, INPUT_PORT);
KDAQ_DIO_PortConfig(card, Channel_P1CH, INPUT_PORT);
//DI operation
KDAQ_DI_ReadPort(card, Channel_P1A, &inputA);
...
KDAQ_Release_Card(card);
    
```

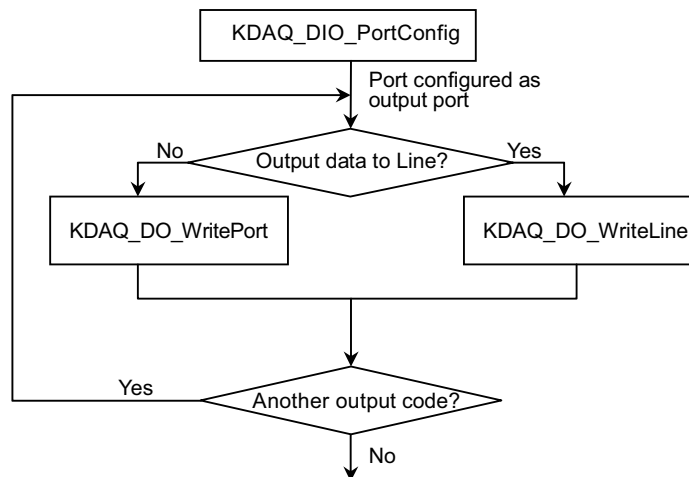
**Digital output programming hints**

KDAQ-DRVR provides one kind of digital output operation — a non-buffered single-point digital output operation. The non-buffered single-point DO uses a software polling method to write data to the device. The programming scheme for this kind of DO operation is described in [One-shot digital output programming](#).

**One-shot digital output programming**

This programming section describes the typical flow of a non-buffered single-point digital output operation. When performing one-shot DO operation for devices whose I/O port can be set as input or output, include port configuration at the beginning of the application. Refer to [Figure A-30](#) and the example code fragment immediately following the figure.

Figure A-30  
**Typical flow of non-buffered single-point digital output operation**



**Example code fragment**

```

card = KDAQ_Register_Card(KPXI_SDAQ_4_2M, card_number);
//port configured
KDAQ_DIO_PortConfig(card ,Channel_P1A, OUTPUT_PORT);
KDAQ_DIO_PortConfig(card, Channel_P1B, OUTPUT_PORT);
KDAQ_DIO_PortConfig(card, Channel_P1CL, OUTPUT_PORT);
KDAQ_DIO_PortConfig(card, Channel_P1CH, OUTPUT_PORT);
//DO operation
KDAQ_DO_WritePort(card, Channel_P1A, outA_value);
...
KDAQ_Release_Card(card);
    
```

## DAQ event message programming hints

DAQ Event Message functions are an efficient way to monitor background data acquisition processes without dedicating foreground processes for status checking. There are two kinds of events: AI/AO operation-completeness notification event and half-buffer-ready notification event.

To receive notification from the KDAQ-DRVR data acquisition process in case of special events, call **KDAQ\_AI\_EventCallback** or **KDAQ\_AO\_EventCallback** to specify an event of interest.

Event notification is done through user-defined callbacks. When a user-specified DAQ event occurs, KDAQ-DRVR calls the user-defined callback. After receiving the message, the user's application can carry out the appropriate task.

If a callback function is called, succeeding events will not be handled until your callback has returned. If the time interval between events is smaller than the time taken for callback function processing, the succeeding events will not be handled. Therefore, do not use this mechanism for frequently occurring event conditions.

### Example code fragment

```
card = KDAQ_Register_Card(KPXI_SDAQ_4_2M, card_number);
...
KDAQ_AI_CH_Config (card, channel, range )
KDAQ_AI_Config (card, 0, KDAQ_AI_TRGMOD_PRE|KDAQ_AI_TRGSRC_ExtD, 0, 0, 0, 1);
// or
// KDAQ_AI_MiddleTrig_Config (card, KDAQ_AI_ADCONVSRC_Int, KDAQ_AI_TRGSRC_ExtD|
KDAQ_AI_TrgPositive, 0, 0, 1);
KDAQ_AI_AsyncDblBufferMode (card, 1); // Double-buffered AI
KDAQ_AI_ContBufferSetup (card, ai_buf, data_size, &BufId);
KDAQ_A_ContBufferSetup (card, ai_buf2, data_size, &BufId);
// Enable half buffer ready event notification
    KDAQ_AI_EventCallback (card, 1, DBEvent, (U32) DB_cbfn );
//Enable AI completeness event notification
KDAQ_AI_EventCallback (card, 1, DAQEnd, (U32) AI_cbfn );
KDAQ_AI_ContScanChannels (card, channel, BufId, data_size/(channel+1),
ScanIntrv, SampIntrv, ASYNCH_OP); or
KDAQ_AI_ContReadChannel(card, channel, BufId, data_size, ScanIntrv, SampIntrv,
ASYNCH_OP)
...
KDAQ_Release_Card(card);

//Half buffer ready call back function
void DB_cbfn()
{
//half buffer is ready
...
}

//AI completeness call back function
void AI_cbfn()
{
//AI is completed
KDAQ_AI_AsyncClear(card, &startPos, &count);
...
}
.
```

## Continuous data transfer in KDAQ-DRVR

The continuous data transfer functions in KDAQ-DRVR input or output blocks of data to or from a plug-in Keithley Instruments PXI DAQ Device. For input operations, KDAQ-DRVR must transfer the incoming data to a buffer in the computer memory. For output operations, KDAQ-DRVR must transfer outgoing data from a buffer in the computer memory to the Keithley Instruments PXI DAQ Device. This chapter describes the mechanism and techniques that KDAQ-DRVR uses for continuous data transfer and the considerations for selecting the continuous data transfer mode (sync. or async., double buffered or not, triggered or non-triggered mode).

### Continuous data transfer mechanism

KDAQ-DRVR uses the DMA controller chip to perform a hardware transfer of the data.

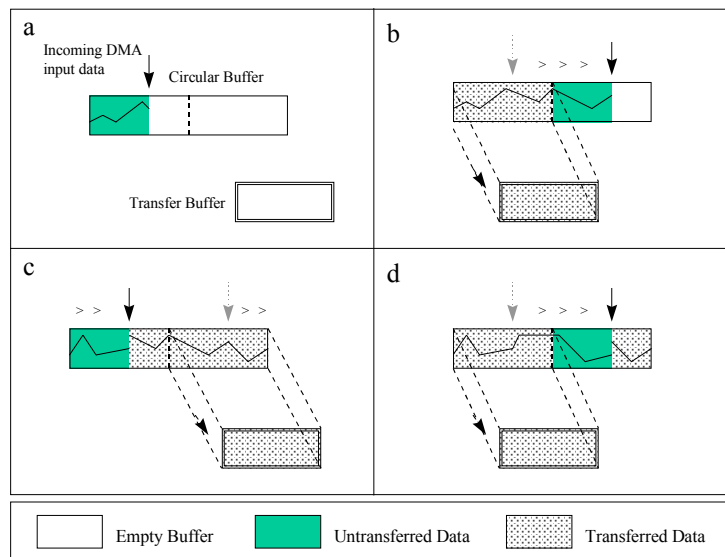
### Double-buffered AI/AO operation

KDAQ-DRVR uses double-buffering techniques in its driver software for continuous input/output of large amounts of data.

#### Double buffer mode principle

The data buffer for double-buffered continuous input operation is a circular logic buffer (logically divided into two equal halves). The double-buffered input begins when the device starts writing data into the first half of the circular buffer (Figure A-31a). After the device begins writing to the second half of the circular buffer, you can process the data in the first half buffer according to application needs (Figure A-31b). After the board has filled the second half of the circular buffer, the board returns to the first half buffer and overwrites the old data. You now can process the second half of the circular buffer (Figure A-31c). The process can be repeated endlessly to provide a continuous stream of data to your application (Figure A-31d).

Figure A-31  
Double buffer mode principle



The KDAQ-DRVR double buffer mode functions were designed according to the principle described above. To enable double buffer mode, when using **KDAQ\_AI\_AsyncDbIBufferMode** or **KDAQ\_AO\_AsyncDbIBufferMode**, any following continuous AI/AO function will perform double-buffered continuous AI/AO. Call **KDAQ\_AI\_AsyncDbIBufferHalfReady** or

**KDAQ\_AO\_AsyncDbIBufferHalfReady** to check if data in the circular buffer is half full and ready for copying to the transfer buffer to occur.

## Single-buffered versus double-buffered data transfer

Single-buffered data transfer is the most common method for continuous data transfer. In single-buffered input operations, a fixed number of samples are acquired at a specified rate and transferred into the user's buffer. After the user's buffer stores the data, the application can analyze, display, or store the data to the hard disk for later processing. Single-buffered operations are relatively simple to implement and can usually take advantage of the full hardware speed of the device. However, the major disadvantage of single-buffered operation is that the maximum amount of data that can be input at any one time is limited to the amount of initially allocated memory allocated in the driver and the amount of free memory available in the computer.

In double-buffered operations, as mentioned above, the data buffer is configured as a circular buffer. Therefore, unlike single-buffered operations, double-buffered operations reuse the same buffer and are able to input or output an infinite number of data points without requiring an infinite amount of memory. However, using double-buffered operation also creates the undesired possibility of data being over-written during the double-buffered data transfer (the device might over-write data before KDAQ-DRVR has copied it to the transfer buffer). Another data over-writing problem occurs when an input device overwrites data that KDAQ-DRVR is simultaneously copying to the transfer buffer. Therefore, the data must be processed by the application at least as fast as the rate at which the device is reading data. For most of the applications, this requirement depends on the speed and efficiency of the computer system and programming language.

Taking these possible undesirable results into account, double buffering might not be practical for high-speed input applications.

## Pre-trigger mode/middle-trigger data acquisition (AI)

A trigger is an event that occurs based on a specified set of conditions. An interrupt mode or DMA-mode analog input operation can use a trigger to determine when the acquisition stops or starts.

KDAQ-DRVR also provides two buffering methods for pre/middle-trigger mode AI-double-buffering and AI-single-buffering. However, the single buffer in pre/middle-trigger mode AI is different from that in non-trigger mode AI. It is a circular buffer (just like that in double buffer mode) but the data stored in the buffer can be processed only when the continuous data reading is completed. The buffer will be reused until the data acquisition operation is completed. Therefore, to protect the data you want to get from being overwritten, the size of the single buffer should be the same as or larger than the amount of data you wish to access. For example, if you want to perform single-buffered middle-trigger AI with KPXI-SDAQ-4-2M, and the amount of data you want to collect before and after the trigger event are 1000 and 3000 respectively, make the size of single buffer at least 4000 in order to get all the data you want to collect. Since the data is handled after the input operation is complete, the desired data loss problem hardly ever occurs.

Since KDAQ-DRVR uses asynchronous AI to perform pre/middle-trigger mode data acquisition, the **SyncMode** of continuous AI should be set as **ASYNCH\_OP**.



---

## KDAQ-DRVR Function Reference

### In this appendix:

Topic	Page
Function description .....	B-2
Data types.....	B-2
Function reference.....	B-2
Status Codes.....	B-94
AI range codes.....	B-95
AI data format.....	B-97
DATA file format .....	B-97
Header.....	B-98
ChannelRange.....	B-99
Data Block .....	B-99

## Function description

This section is provided as a function reference. It contains a detailed description of KDAQ-DRVR functions and includes information on KDAQ-DRVR [Data types](#) as well as a [KDAQ-DRVR Function reference](#) (functions are arranged alphabetically in the reference). Syntax is provided for Microsoft C/C++, Borland C++, and Visual Basic.

### Data types

[Table B-1](#) contains data types defined in **hdaqdrvr.h**. These data types are used by the KDAQ-DRVR library. It is recommended these data types are used in your application programs. [Table B-1](#) contains data type names, ranges, and the corresponding data types for C/C++, and Visual Basic.

**NOTE** The data types in [Table B-1](#) are defined in **hdaqdrvr.h**, but are not defined in **kdaqdrvr.bas** or **kdaqdrvr.pas** (for **.bas** and **.pas** definition files, the table is provided only as a reference).

Table B-1  
Suggested data types

Type Name	Description	Range	Type	
			C/C++ ( for 32-bit compiler)	Visual Basic
U8	8-bit ASCII character	0 to 255	unsigned char	Byte
I16	16-bit signed integer	-32768 to 32767	short	Integer
U16	16-bit unsigned integer	0 to 65535	unsigned short	Not supported by this type, use the signed integer (I16) instead
I32	32-bit signed integer	-2147483648 to 2147483647	long	Long
U32	32-bit unsigned integer	0 to 4294967295	unsigned long	Not supported by this type, use the signed long integer (I32) instead
F32	32-bit single-precision floating-point	-3.402823E38 to 3.402823E38	float	Single
F64	64-bit double-precision floating-point	-1.797683134862315E308 to 1.797683134862315E309	double	Double

### Function reference

#### KDAQ\_AI\_AsyncCheck

**Description** Checks current status of the asynchronous analog input operation.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AI_AsyncCheck (U16 CardNumber, BOOLEAN *Stopped,
    U32 *AccessCnt)
```

**Visual Basic**

```
KDAQ_AI_AsyncCheck (ByVal CardNumber As Integer,
    Stopped As Byte, AccessCnt As Long) As Integer
```

**Parameters** **CardNumber:** The identification of the card performing the asynchronous operation.

**Stopped:** The status of the asynchronous analog input operation. If **Stopped** = TRUE, the analog input operation has stopped. This is caused by either the number of A/D conversions indicated in the call that initiated the input operation has completed or an error has occurred. If **Stopped** = FALSE, the operation has not completed (constants TRUE and FALSE are defined in **kdaqdrv.h**).

**AccessCnt :** As long as **pre-trigger** or **middle trigger** mode of AI acquisition is not enabled, **AccessCnt** returns the number of A/D data points that have been transferred at the time of calling **KDAQ\_AI\_AsyncCheck()**.

If **pre-trigger** or **middle trigger** mode of AI and double-buffered mode is enabled, **AccessCnt** returns the position following the last position A/D data was stored in the circular buffer at the time **KDAQ\_AI\_AsyncCheck()** was called.

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport

### KDAQ\_AI\_AsyncClear

**Description** Stops the asynchronous analog input operation.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AI_AsyncClear (U16 CardNumber, U32 *StartPos,
    U32 *AccessCnt)
```

**Visual Basic**

```
KDAQ_AI_AsyncClear (ByVal CardNumber As Integer,
    StartPos As Long, AccessCnt As Long) As Integer
```

**Parameters** **CardNumber:** The card id of the card that performs the asynchronous operation.

**StartPos:** If trigger acquisition mode is not used, **StartPos** is zero. If **pre-trigger** or **middle trigger** acquisition mode of AI is used, **StartPos** returns the position of the first AD data in the data buffer at the time of calling **KDAQ\_AI\_AsyncClear()**.

**AccessCnt:** In the condition that the **pre-trigger** or **middle trigger** acquisition mode is not used, **AccessCnt** returns the number of A/D data points that have been transferred at the time of calling **KDAQ\_AI\_AsyncClear()**.

If **double-buffered** mode is enabled, **AccessCnt** returns the next position after the position the last A/D data is stored in the circular buffer.

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport

### KDAQ\_AI\_AsyncDbIBufferHalfReady

**Description** Checks if the next half-buffer of data (in the circular buffer) is ready for transfer during an asynchronous double-buffered analog input operation.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax****Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AI_AsyncDblBufferHalfReady (U16 CardNumber,
    BOOLEAN *HalfReady, BOOLEAN *StopFlag)
```

**Visual Basic**

```
KDAQ_AI_AsyncDblBufferHalfReady (ByVal CardNumber As Integer,
    HalfReady As Byte, StopFlag As Byte) As Integer
```

**Parameters**

**CardNumber** : The identification of the card performing the asynchronous double-buffered operation.

**HalfReady** : Whether the next half buffer of data is available (constants TRUE and FALSE are defined in **kdaqdrv.h**).

**StopFlag** : Status of the asynchronous analog input operation. If **StopFlag** = TRUE, the analog input operation has stopped. If **StopFlag** = FALSE, the operation is not yet complete (constants TRUE and FALSE are defined in **kdaqdrv.h**).

**Return Value**

NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport

**KDAQ\_AI\_AsyncDblBufferHandled****Description**

Notifies KDAQ-DRVR that the ready buffer has been handled in user application.

For KDAQ-DRVR, the data is transferred (through DMA) to the user's buffer directly. Therefore, if the next half-buffer of data is ready (using KDAQ\_AI\_AsyncDblBufferHalfReady to check the ready status), the data in the ready buffer can be handled directly and don't needed to be copied to another transfer buffer. This mechanism eliminates the time taken for memory copy and another memory space for data transfer; however, KDAQ-DRVR couldn't know if the data in the ready buffer have been handled (in user application). If the data is handled, the user application needs an interface to notify KDAQ-DRVR this information. The function KDAQ\_AI\_AsyncDblBufferHandled is used to for this purpose.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax****Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AI_AsyncDblBufferHandled (U16 CardNumber)
```

**Visual Basic**

```
KDAQ_AI_AsyncDblBufferHandled (ByVal CardNumber As Integer)
    As Integer
```

**Parameters**

**CardNumber**: The card id of the card that double-buffered mode is to be set on.

**Return Value**

NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport

### KDAQ\_AI\_AsyncDblBufferMode

**Description** Enables or disables double-buffered data acquisition mode.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AI_AsyncDblBufferMode (U16 CardNumber,
    BOOLEAN Enable)
```

**Visual Basic**

```
KDAQ_AI_AsyncDblBufferMode (ByVal CardNumber As Integer,
    ByVal Enable As Byte) As Integer
```

**Parameters** **CardNumber:** The card id of the card that double-buffered mode is to be set on.  
**Enable:** Whether the double-buffered mode is enabled or not.

TRUE: double-buffered mode is enabled.  
 FALSE: double-buffered mode is disabled.

Constants TRUE and FALSE are defined in **kdaqdrv.h**.

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport

### KDAQ\_AI\_AsyncDblBufferOverrun

**Description** Checks or clears overrun status of the double-buffered analog input operation.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AI_AsyncDblBufferOverrun (U16 CardNumber, U16 op,
    U16 *overrunFlag)
```

**Visual Basic**

```
KDAQ_AI_AsyncDblBufferOverrun (ByVal CardNumber As Integer,
    ByVal op As Integer, overrunFlag As Integer) As Integer
```

**Parameters** **CardNumber:** The card id of the card that double-buffered mode is to be set on.  
**op:** check/clear overrun status/flag.  
 0: check the overrun status.  
 1: clear the overrun flag.

**overrunFlag:** returned overrun status  
 0: no overrun occurs.  
 1: overrun occurs.

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport

## KDAQ\_AI\_AsyncDbIBufferToFile

**Description** If the continuous AI function is *KDAQ\_AI\_Cont ReadChannelToFile*, *KDAQ\_AI\_ContReadMultiChannelsToFile*, *KDAQ\_AI\_ContScanChannelsToFile* or *KDAQ\_AI\_ContMuxScanToFile* call this function to log the data of the circular buffer into a disk file.

This function is supported by the following mode: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AI_AsyncDbIBufferToFile (U16 CardNumber)
```

### Visual Basic

```
KDAQ_AI_AsyncDbIBufferToFile (ByVal CardNumber As Integer)
    As Integer
```

**Parameters** *CardNumber*: The card id of the card that double-buffered mode is to be set on.

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport

## KDAQ\_AI\_AsyncReTrigNextReady

**Description** Checks whether the data associated to the next trigger signal is ready during an asynchronous re-triggered analog input operation. There are some restrictions to using this function:

- a. *KDAQ\_AI\_Config* has to be called **prior to** *KDAQ\_AI\_ContBufferSetup*.
- b. Asynchronous mode should be used with *ASYNCH\_OP* for continuous AI operation.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AI_AsyncReTrigNextReady (U16 wCardNumber,
    BOOLEAN *trgReady, BOOLEAN *StopFlag, U16 *RdyTrigCnt)
```

### Visual Basic

```
KDAQ_AI_AsyncReTrigNextReady (ByVal CardNumber As Integer,
    trgReady As Byte, StopFlag As Byte, RdyTrigCnt As Integer)
    As Integer
```

**Parameters** *CardNumber*: The card id of the card that performs the asynchronous re-trigger operation.

*trgReady*: Whether the data associated to the next trigger signal is available. (constants TRUE and FALSE are defined in *kdaqdrv.h*)

*StopFlag*: Whether the asynchronous analog input operation has completed. If *StopFlag* = TRUE, the analog input operation has stopped. If *StopFlag* = FALSE,

the operation is not yet complete. (constants TRUE and FALSE are defined in kdaqdrvr.h)

**RdyTrigCnt:** If retrigger count is **defined**, this argument returns the count of trigger signals that happened. If the retrigger count is **infinite**, the argument "RdyTrigCnt" returns the index of the buffer storing the data of the most recently generated trigger signal trigger.

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport

### KDAQ\_AI\_CH\_Config

**Description** Informs KDAQ-DRVR library of the AI range selected for the specified channel of the card with card ID *CardNumber*. After the function "KDAQ\_Register\_Card" is called, all of the analog input channels are configured as AD\_B\_10\_V (for KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-AO-4-1M, KPXI-AO-8-1M) or AD\_B\_10\_V with AI\_RSE (for KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, and KPXI-DAQ-96-3M) by default. If you wish to perform the operation with the default settings, it is not necessary to call this function to configure the channel(s) again. Otherwise, this function has to be called to program the device for the settings you want before calling the function to perform analog input operation.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AI_CH_Config (U16 wCardNumber, U16 wChannel,
    U16 wAdRange_RefGnd)
```

#### Visual Basic

```
KDAQ_AI_CH_Config (ByVal CardNumber As Integer,
    ByVal Channel As Integer, ByVal AdRange_RefGnd As Integer)
    As Integer
```

**Parameters** **CardNumber:** The card id of the card to perform this operation.

**Channel:** The A/D channel wished to do the channel setting.  
Valid values:

- KPXI-SDAQ-4-2M : 0 through 3 or All\_Channels (-1)
- KPXI-SDAQ-4-500K : 0 through 3 or All\_Channels (-1)
- KPXI-DAQ-64-3M : 0 through 63 or All\_Channels (-1)
- KPXI-DAQ-64-500K : 0 through 63 or All\_Channels (-1)
- KPXI-DAQ-64-250K : 0 through 63 or All\_Channels (-1)
- KPXI-DAQ-96-3M : 0 through 95 or All\_Channels (-1)
- KPXI-AO-4-1M : 0 through 7 or All\_Channels (-1)
- KPXI-AO-8-1M : 0 through 3 or All\_Channels (-1)

**Parameters** *AdRange\_RefGnd*: The settings for analog input channel. This argument is an integer expression formed from one or more of the manifest constants defined in **kdaqdrv.h**. There are two groups of constants:

#### A/D range Selection

We define some constants to represent various A/D input ranges in **kdaqdrv.h**. Please refer to the [AI range codes](#), for the valid range values. The default setting is AD\_B\_10\_V.

**A/D reference Ground Selection** (*only available for for KPXI-DAQ-64-3M/ KPXI-DAQ-64-500K/KPXI-DAQ-64-250K/KPXI-DAQ-96-3M*)

KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-AO-4-1M, KPXI-AO-8-1M: 0

KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M:

*AI\_RSE*: Referenced single ended mode ( 64chs common to ground system on board) (default value)

*AI\_DIFF*: Differential mode

*AI\_NRSE*: Non-referenced single ended mode (64 channels common to AISENSE pin)

When two or more constants are used to form the *AdRange\_RefGnd* argument, the constants are combined with the bit wise-OR operator(`|`).

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport

## KDAQ\_AI\_Config

**Description** Informs KDAQ-DRVR library of the trigger source, trigger mode and trigger properties for the Keithley PXI DAQ device with card ID *CardNumber*.

After the function **Register\_Card** is called, the device is configured as the following by default:

A/D conversion source: KDAQ\_AI\_ADCONVSRC\_Int

A/D trigger mode : KDAQ\_AI\_TRGMOD\_POST

A/D trigger source : KDAQ\_AI\_TRGSRC\_SOFT

Auto reset buffer: Enabled (AutoResetBuf : TRUE)

If you wish to perform the device with the default settings, it is not necessary to call this function to make the configuration again. Otherwise, this function has to be called before calling function to perform continuous analog input operation.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AI_Config (U16 wCardNumber, U16 ConfigCtrl,
    U32 TrigCtrl, U32 MidOrDlyScans, U16 MCnt, U16 ReTrgCnt,
    BOOLEAN AutoResetBuf)
```



**Visual Basic**

KDAQ\_AI\_Config (ByVal CardNumber As Integer, ByVal ConfigCtrl As Integer, ByVal TrigCtrl As Long, ByVal MidOrDlyScans As Long, ByVal MCnt As Integer, ByVal ReTrgCnt As Integer, ByVal AutoResetBuf As Byte) As Integer

**Parameters**

**CardNumber:** The card id of the card to perform this operation.

**ConfigCtrl:** The setting for A/D configuration control. This argument is an integer expression formed from one or more of the manifest constants defined in **kdaqdrv.h**. There are two groups of constants:

**(1) A/D Conversion Source Selection**

KDAQ\_AI\_ADCONVSRC\_Int : Internal timer (default)

KDAQ\_AI\_ADCONVSRC\_AFI0: From AFI0 pin

KDAQ\_AI\_ADCONVSRC\_SSI: From SSI source

KDAQ\_AI\_ADCONVSRC\_AFI1: From AFI1 pin (**only available for KPXI-AO-4-1M and KPXI-AO-8-1M**)

**(2) A/D Delay Counter Source Selection (only available for KPXI-AO-4-1M and KPXI-AO-8-1M)**

KDAQ\_AI\_DTSRC\_Int : Internal timer (default)

KDAQ\_AI\_DTSRC\_AFI1: From AFI1 pin

KDAQ\_AI\_DTSRC\_GPTC0: From GPTC0\_OUT

KDAQ\_AI\_DTSRC\_GPTC1: From GPTC1\_OUT

When two or more constants are used to form the *ConfigCtrl* argument, the constants are combined with the bit wise-OR operator(**|**).

**TrigCtrl:** The setting for A/D Trigger control. This argument is an integer expression formed from one or more of the manifest constants defined in **kdaqdrv.h**. There are seven groups of constants:

**(1) Trigger Source Selection**

KDAQ\_AI\_TRGSRC\_SOFT : software (default)

KDAQ\_AI\_TRGSRC\_ANA : From analog trigger pin

KDAQ\_AI\_TRGSRC\_ExtD: From external digital trigger pin

KDAQ\_AI\_TRSRC\_SSI : From SSI source

**(2) Trigger Mode Selection**

KDAQ\_AI\_TRGMOD\_POST : Post Trigger Mode (default)

KDAQ\_AI\_TRGMOD\_DELAY : Delay Trigger Mode

KDAQ\_AI\_TRGMOD\_PRE : Pre-Trigger Mode

KDAQ\_AI\_TRGMOD\_MIDL : Middle-Trigger Mode

**(3) Delay Source Selection (only available for Delay Trigger Mode)**

KDAQ\_AI\_Dly1InSamples: delay in samples

KDAQ\_AI\_Dly1InTimebase: delay in time base (default)

**(4) Re-Trigger Mode Enable (only available for Delay and Post Trigger Mode)**

KDAQ\_AI\_ReTrigEn: Re-trigger in an acquisition is enabled

**(5) MCounter Enable (only available for Pre- and Middle Trigger Mode)**

This constant is only valid for Pre-trigger and Middle trigger mode

*KDAQ\_AI\_MCounterEn*: Mcounter is enabled and then the trigger signal is ignored before M terminal count is reached.

**(6) External Digital Trigger Polarity**

*KDAQ\_AI\_TrgPositive*: Trigger positive edge active (default)

*KDAQ\_AI\_TrgNegative*: Trigger negative edge active

When two or more constants are used to form the *TrigCtrl* argument, the constants are combined with the bit wise OR operator(`|`).

***MidOrDlyScans***: This argument is only valid for Middle trigger and Delay trigger mode.

For *Middle trigger*, *MidOrDlyScans* indicates the number of data will be accessed after a specific trigger event. The valid value range of *MidOrDlyScans* for middle trigger is 0 through 16777215

For *Delay trigger*, *MidOrDlyScans* indicates the number of data or timer ticks that will be ignored after a specific trigger event. The valid value range of *DlyScans* for delay trigger is 0 through 65535.

***MCnt***: The counter value of MCounter. The valid value range of *MCnt* is 0 through 65535. This argument is only valid for pre-trigger and Middle trigger mode.

***ReTrgCnt***: The accepted trigger times in an acquisition. The valid value range of *ReTrgCnt* is 0 through 65535. If the value of *ReTrgCnt* is 0, the AI operation will be triggered **infinitely**. This argument is only valid for Delay trigger and Post trigger mode.

**NOTE** To enable infinite re-trigger mode of continuous AI, Calling `KDAQ_AI_Config` with `KDAQ_AI_ReTrigEn` and zero value of `ReTrgCnt`.

**AutoResetBuf:**

FALSE: The AI buffers set by function `KDAQ_AI_ContBufferSetup` are retained and must call function `KDAQ_AI_ContBufferReset` to reset the buffer

TRUE: The AI buffers set by function `KDAQ_AI_ContBufferSetup` are reset automatically by the driver while the AI operation is finished

**NOTE** If `Mcounter` is enabled, the `ReadScans` parameter of continuous AI functions `KDAQ_AI_ContXXXX` has to be equal to `MidOrDlyScans+MCnt`.

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport

**KDAQ\_AI\_ContBufferReset**

**Description** This function reset all the buffers set by function `KDAQ_AI_ContBufferSetup` for continuous analog input. The function has to be called if the data buffers won't be used.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

`I16 KDAQ_AI_ContBufferReset (U16 wCardNumber)`

**Visual Basic**

`KDAQ_AI_ContBufferReset (ByVal CardNumber As Integer) As Integer`

**Parameters** *CardNumber*: The card id of the card to perform this operation.

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorTransferCountTooLarge , ErrorContloNotAllowed

**KDAQ\_AI\_ContBufferSetup**

**Description** This function sets up the buffer for continuous analog input. The function has to be called repeatedly to setup all of the data buffers (at most 2 buffers). **For double buffer mode and infinite re-trigger mode of continuous AI**, call `KDAQ_AI_ContBufferSetup` **twice** to setup the ring buffer to store the data.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

`I16 KDAQ_AI_ContBufferSetup (U16 wCardNumber, void *pwBuffer, U32 dwReadCount, U16 *BufferId)`

**Visual Basic**

`KDAQ_AI_ContBufferSetup (ByVal CardNumber As Integer, Buffer As Any, ByVal ReadCount As Long, BufferId As Integer) As Integer`

- Parameters**
- CardNumber:** The card id of the card to perform this operation.
  - Buffer:** The starting address of the memory to contain the input data.
  - ReadCount:** The size (in samples) of the buffer and its value must be even.
  - BufferId:** Returns the index of the buffer currently set up.
- Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorTransferCountTooLarge , ErrorContloNotAllowed

## KDAQ\_AI\_ContMuxScan

**Description** This function initializes the Channel-Gain Queue to point to the start of the scan sequence as specified by *KDAQ\_AI\_MuxScanSetup* and starts a multiple-channel scanned data acquisition operation. This function is only available for **Multiplexed AD card** (e.g. KPXI-DAQ-64-500K).

This function is supported by the following models: KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AI_ContMuxScan (U16 wCardNumber, U16 BufId,
    U32 ReadScans, U32 ScanIntrv, U32 SampIntrv,
    U16 wSyncMode)
```

### Visual Basic

```
Function KDAQ_AI_ContMuxScan (ByVal CardNumber As Integer,
    ByVal BufId As Integer, ByVal ReadScans As Long,
    ByVal ScanIntrv As Long, ByVal SampIntrv As Long,
    ByVal SyncMode As Integer) As Integer
```

**Parameters** **CardNumber:** The card id of the card to perform this operation.

**BufId:** The buffer ID (returned from function *KDAQ\_AI\_ContBufferSetup*) of the buffer containing the acquired data. The size of the buffer with buffer id of *BufId* must have a length equal to the value of parameter *ScanCount* \* (*number of channels per scan*). If double-buffered mode is enabled, The starting buffer id should be 0. You can ignore this argument. Please refer to [AI data format](#) for the data format in the buffer with *BufId*.

**ReadScans:** If double-buffered mode is disabled, the total number of scans to be performed. For double-buffered acquisition, *ReadScans* is the size (in samples) allocated for each channel in the circular buffer. The valid range of the value is 2 through 16777215. This value must be a multiple of 2.

**NOTE** If *Mcounter* is enabled, set **ReadScans** to a value equal to **MidOrDlyScans+MCnt**.

**ScanIntrv:** The length of the scan interval (that is, the counter value between the initiation of each scan sequence). The scan rate will be *TimeBase/ScanIntrv*. The value of *TimeBase* depends on the card type.

If the timer base is from *external*, the valid range of the value is 8 through 16777215. If the timer base is *Internal timer*, the valid range of the value is as follows:

KPXI-DAQ-64-3M : 14 through 16777215  
 KPXI-DAQ-64-500K : 80 through 16777215

KPXI-DAQ-64-250K : 160 through 16777215  
 KPXI-DAQ-96-3M : 14 through 16777215

**Samplntrv:** The length of the sample interval (that is, the counter value between each A/D conversion within a scan sequence). The A/D conversion rate will be  $TimeBase/Samplntrv$ . The value of *TimeBase* depends on the card type.

If the timer base is from *external*, the valid range of the value is 8 through 65535.

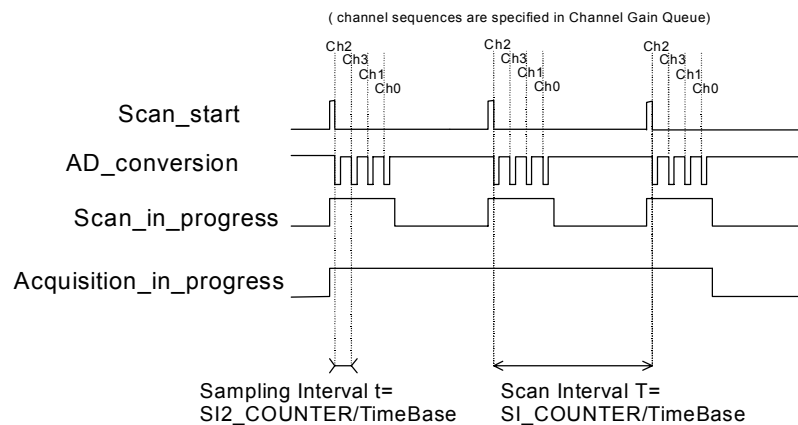
If the timer base is *Internal timer*, the valid range of the value is as follows:

KPXI-DAQ-64-3M : 14 through 65535  
 KPXI-DAQ-64-500K : 80 through 65535  
 KPXI-DAQ-64-250K : 160 through 65535  
 KPXI-DAQ-96-3M : 14 through 65535

See [Figure B-1](#) for an example.

Figure B-1  
**Scan timing example**

3 Scans, 4 Samples per scan  
 (PSC\_Counter=3, NumChan\_Counter=4)



**NOTE** If the card is **Simultaneous AD card** (e.g., KPXI-SDAQ-4-2M), the parameter **Samplntrv** does not have an effect.

**SyncMode:** Whether this operation is performed synchronously or asynchronously. If pre-/middle trigger mode is enabled by calling **KDAQ\_AI\_Config()**, this operation should be performed **asynchronously**.

Valid values:

SYNCH\_OP:synchronous A/D conversion, that is, the function does not return until the A/D operation completes.

ASYNCH\_OP:asynchronous A/D conversion

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel, ErrorInvalidSampleRate, ErrorInvalidAdRange, ErrorTransferCountTooLarge, ErrorContIoNotAllowed

## KDAQ\_AI\_ContMuxScanToFile

**Description** Initializes the Channel-Gain Queue to point to the start of the scan sequence as specified by *KDAQ\_AI\_MuxScanSetup*, starts a multiple-channel scanned data acquisition operation and saves the acquired data in a disk file. The data is written to disk in binary format, with the lower byte first (little endian). Please refer to [DATA file format](#) for the data file structure and [AI data format](#) for the format of the data in the data file. This function takes advantage of the Keithley PXI DAQ channel-gain that can be set separately for each channel to perform multi-channel/gain analog input. This function is only available for a **Multiplexed AD card** (e.g. KPXI-DAQ-64-500K).

This function is supported by the following models: KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AI_ContMuxScanToFile (U16 wCardNumber, U16 BufId,
    U8 *fileName, U32 ReadScans, U32 ScanIntrv, U32 SampIntrv,
    U16 wSyncMode)
```

### Visual Basic

```
KDAQ_AI_ContMuxScanToFile (ByVal CardNumber As Integer,
    ByVal BufId As Integer, ByVal FileName As String,
    ByVal ReadScans As Long, ByVal ScanIntrv As Long,
    ByVal SampIntrv As Long, ByVal SyncMode As Integer)
    As Integer
```

**Parameters** **CardNumber:** The card id of the card to perform this operation.

**BufId:** The buffer ID (returned from function *KDAQ\_AI\_ContBufferSetup*) of the buffer containing the acquired data. The size of the buffer with buffer id of *BufId* must have a length equal to the value of parameter *ReadScans* \* (*number of channels per scan*). If double-buffered mode is enabled, The starting buffer id should be 0. You can ignore this argument. Please refer to [AI data format](#) for the data format in the buffer with *BufId*.

**FileName:** Name of data file which stores the acquired data

**ReadScans:** If double-buffered mode is disabled, the total number of scans to be performed. For double-buffered acquisition, *ReadScans* is the size (in samples) allocated for each channel in the circular buffer. The valid range of the value is 2 through 16777215. This value must be a multiple of 2.

**NOTE** If *Mcounter* is enabled, the *ReadScans* has to be equal to *MidOrDlyScans+MCnt*.

**ScanIntrv:** The length of the scan interval (that is, the counter value between the initiation of each scan sequence). The scan rate will be *TimeBase/ScanIntrv*. The value of *TimeBase* depends on the card type.

If the timer base is from *external*, the valid range of the value is 8 through 16777215. If the timer base is *Internal timer*, the valid range of the value is as follows:

KPXI-DAQ-64-3M : 14 through 16777215  
 KPXI-DAQ-64-500K : 80 through 16777215

KPXI-DAQ-64-250K : 160 through 16777215  
 KPXI-DAQ-96-3M : 14 through 16777215

**Samplntrv:** The length of the sample interval (that is, the counter value between each A/D conversion within a scan sequence). The A/D conversion rate will be  $TimeBase/Samplntrv$ . The value of *TimeBase* depends on the card type.

If the timer base is from *external*, the valid range of the value is 8 through 65535.

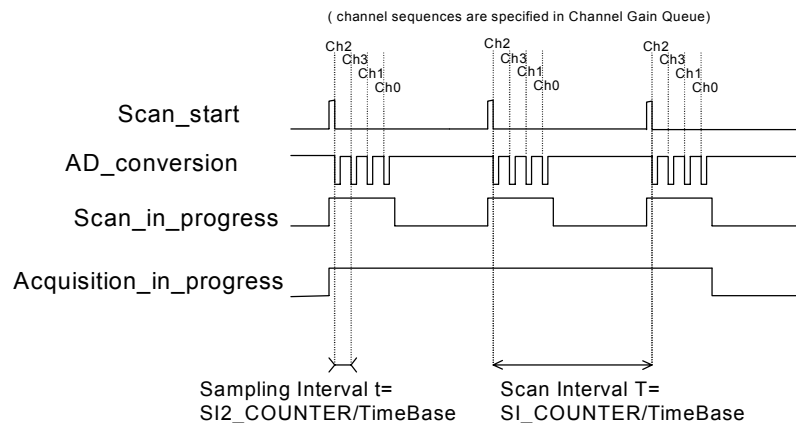
If the timer base is *Internal timer*, the valid range of the value is as follows:

KPXI-DAQ-64-3M : 14 through 65535  
 KPXI-DAQ-64-500K : 80 through 65535  
 KPXI-DAQ-64-250K : 160 through 65535  
 KPXI-DAQ-96-3M : 14 through 65535

See [Figure B-2](#) for example.

Figure B-2  
**Scan timing example**

3 Scans, 4 Samples per scan  
 (PSC\_Counter=3, NumChan\_Counter=4)



**NOTE** If the card is a **Simultaneous AD card** (e.g. *KPXI-SDAQ-4-2M*), the parameter **Samplntrv** is of no use.

**SyncMode:** Whether this operation is performed synchronously or asynchronously. If any trigger mode is enabled by calling `KDAQ_AI_Config()`, this operation should be performed **asynchronously**. Valid values:

*SYNCH\_OP*: synchronous A/D conversion, that is, the function does not return until the A/D operation completes.

*ASYNCH\_OP*: asynchronous A/D conversion

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel, ErrorInvalidSampleRate, ErrorInvalidAdRange, ErrorTransferCountTooLarge, ErrorContIoNotAllowed, ErrorOpenFile

## KDAQ\_AI\_ContReadChannel

**Description** This function performs continuous A/D conversions on the specified analog input channel at a rate as close as possible to the rate you specified.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AI_ContReadChannel (U16 CardNumber, U16 Channel,
    U16 BufId, U32 ReadScans, U32 ScanIntrv, U32 SampIntrv,
    U16 SyncMode)
```

### Visual Basic

```
KDAQ_AI_ContReadChannel (ByVal CardNumber As Integer,
    ByVal Channel As Integer, ByVal BufId As Integer,
    ByVal ReadScans As Long, ByVal ScanIntrv As Long,
    ByVal SampIntrv As Long, ByVal SyncMode As Integer)
As Integer
```

**Parameters** **CardNumber:** The card id of the card to perform this operation.

**Channel:** Analog input channel number

Range: 0 through 3 for KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-AO-8-1M

Range: 0 through 7 for KPXI-AO-4-1M

Range: 0 through 63 for KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K

Range: 0 through 95 for KPXI-DAQ-96-3M

**BufId:** The buffer ID (returned from function *KDAQ\_AI\_ContBufferSetup*) of the buffer containing the acquired data. The size of the buffer with buffer id of *BufId* must have a length equal to the value of parameter *ReadScans*. If double-buffered mode is enabled, The starting buffer id should be 0. You can ignore this argument. Please refer to [AI data format](#) for the data format in the buffer with *BufId*.

**ReadScans:** If double-buffered mode is disabled, the total number of scans to be performed. For double-buffered acquisition, *ReadScans* is the size (in samples) allocated for each channel in the circular buffer. The valid range of the value is 2 through 16777215. This value must be a multiple of 2.

**NOTE** If *Mcounter* is enabled, the *ReadScans* has to be equal to *MidOrDlyScans+MCnt*.

**ScanIntrv:** The length of the scan interval (that is, the counter value between the initiation of each scan sequence). The scan rate will be *TimeBase/ScanIntrv*. The value of *TimeBase* depends on the card type.



If the timer base is from *external*, the valid range of the value is 8 through 16777215. If the timer base is *Internal timer*, the valid range of the value is as follows:

- KPXI-SDAQ-4-2M : 20 through 16777215
- KPXI-SDAQ-4-500K : 80 through 16777215
- KPXI-DAQ-64-3M : 14 through 16777215
- KPXI-DAQ-64-500K : 80 through 16777215
- KPXI-DAQ-64-250K : 160 through 16777215
- KPXI-DAQ-96-3M : 14 through 16777215
- KPXI-AO-4-1M / KPXI-AO-8-1M: 100 through 16777215

**Samplntrv:** The length of the sample interval (that is, the counter value between each A/D conversion within a scan sequence). The A/D conversion rate will be *TimeBase/Samplntrv*. The value of *TimeBase* depends on the card type.

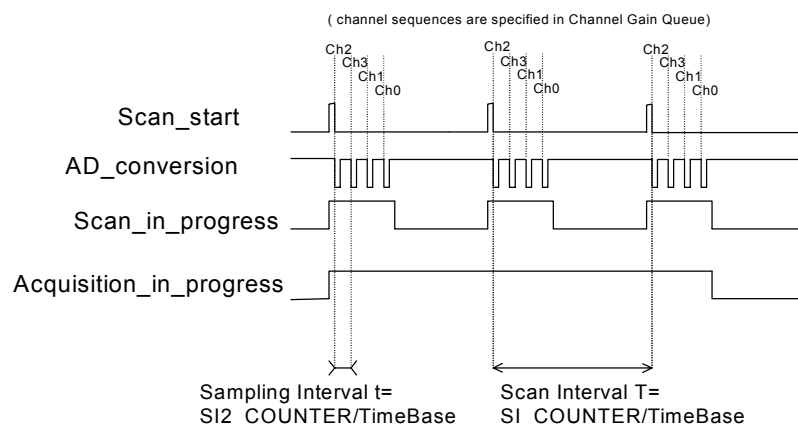
If the timer base is from *external*, the valid range of the value is 8 through 65535.

If the timer base is *Internal timer*, the valid range of the value is as follows:

- KPXI-SDAQ-4-2M : invalid
- KPXI-SDAQ-4-500K : invalid
- KPXI-DAQ-64-3M : 14 through 65535
- KPXI-DAQ-64-500K : 80 through 65535
- KPXI-DAQ-64-250K : 160 through 65535
- KPXI-DAQ-96-3M : 14 through 65535
- KPXI-AO-4-1M / KPXI-AO-8-1M: 100 through 16777215

Figure B-3  
Scan timing example

3 Scans, 4 Samples per scan  
(PSC\_Counter=3, NumChan\_Counter=4)



**NOTE** If the card is a **Simultaneous AD card** (e.g. KPXI-SDAQ-4-2M), the parameter **Samplntrv** is of no use.

**SyncMode:** Whether this operation is performed synchronously or asynchronously. If pre-/middle trigger mode is enabled by calling

**KDAQ\_AI\_Config()**, this operation should be performed *asynchronously*. Valid values:

**SYNCH\_OP**: synchronous A/D conversion, that is, the function does not return until the A/D operation completes.

**ASYNCH\_OP**: asynchronous A/D conversion

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel, ErrorInvalidAdRange, ErrorTransferCountTooLarge, ErrorContIoNotAllowed, ErrorInvalidSampleRate

## KDAQ\_AI\_ContReadChannelToFile

**Description** This function performs continuous A/D conversions on the specified analog input channel at a rate as close as possible to the rate you specified and saves the acquired data in a disk file. The data is written to disk in binary format, with the lower byte first (little endian). Please refer to [DATA file format](#) for the data file structure and [AI data format](#), for the format of the data in the data file.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AI_ContReadChannelToFile (U16 CardNumber,
    U16 Channel, U16 BufId, U8 *FileName, U32 ReadScans,
    U32 ScanIntrv, U32 SampIntrv, U16 SyncMode)
```

### Visual Basic

```
KDAQ_AI_ContReadChannelToFile (ByVal CardNumber As Integer,
    ByVal Channel As Integer, ByVal BufId As Integer,
    ByVal FileName As String, ByVal ReadScans As Long,
    ByVal ScanIntrv As Long, ByVal SampIntrv As Long,
    ByVal SyncMode As Integer) As Integer
```

**Parameters** **CardNumber**: The card id of the card to perform this operation.

**Channel**: Analog input channel number

Range: 0 through 3 for KPXI-SDAQ-4-2M, KPXI-SDAQ-64-500K, KPXI-AO-8-1M

Range: 0 through 7 for KPXI-AO-4-1M

Range: 0 through 63 for KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K

Range: 0 through 95 for KPXI-DAQ-96-3M

**BufId**: The buffer ID (returned from function *KDAQ\_AI\_ContBufferSetup*) of the buffer containing the acquired data. The size of the buffer with buffer id of *BufId* must have a length equal to the value of parameter *ReadScans*. If double-buffered mode is enabled, The starting buffer id should be 0. You can

ignore this argument. Please refer to [AI data format](#) for the data format in the buffer with *BufId*.

**FileName:** Name of data file which stores the acquired data

**ReadScans:** If double-buffered mode is disabled, the total number of scans to be performed. For double-buffered acquisition, *ReadScans* is the size (in samples) allocated for each channel in the circular buffer. The valid range of the value is 2 through 16777215. This value must be a multiple of 2.

**NOTE** If *Mcounter* is enabled, the *ReadScans* has to be equal to *MidOrDlyScans+MCnt*.

**ScanIntrv:** The length of the scan interval (that is, the counter value between the initiation of each scan sequence). The scan rate will be *TimeBase/ScanIntrv*. The value of *TimeBase* depends on the card type.

If the timer base is from *external*, the valid range of the value is 8 through 16777215. If the timer base is *Internal timer*, the valid range of the value is as follows:

KPXI-SDAQ-4-2M : 20 through 16777215  
 KPXI-SDAQ-4-500K : 80 through 16777215  
 KPXI-DAQ-64-3M : 14 through 16777215  
 KPXI-DAQ-64-500K : 80 through 16777215  
 KPXI-DAQ-64-250K : 160 through 16777215  
 KPXI-DAQ-96-3M : 14 through 16777215  
 KPXI-AO-4-1M / KPXI-AO-8-1M: 100 through 16777215

**SamplIntrv:** The length of the sample interval (that is, the counter value between each A/D conversion within a scan sequence). The A/D conversion rate will be *TimeBase/SamplIntrv*. The value of *TimeBase* depends on the card type.

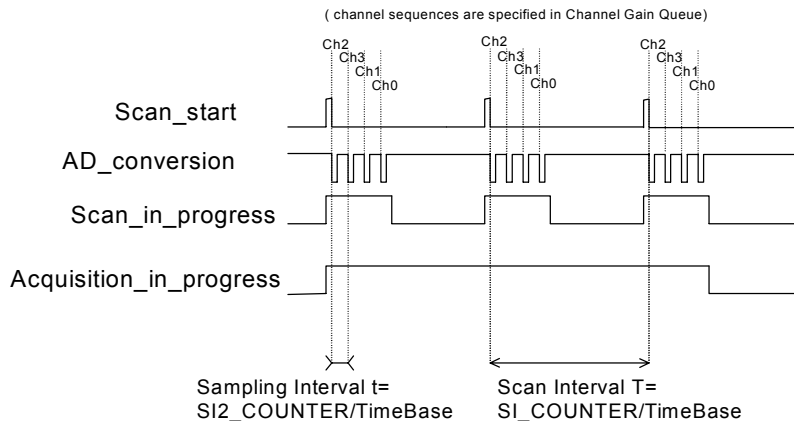
If the timer base is from *external*, the valid range of the value is 8 through 65535.

If the timer base is *Internal timer*, the valid range of the value is as follows:

KPXI-SDAQ-4-2M : invalid  
 KPXI-SDAQ-4-500K : invalid  
 KPXI-DAQ-64-3M : 14 through 65535  
 KPXI-DAQ-64-500K : 80 through 65535  
 KPXI-DAQ-64-250K : 160 through 65535  
 KPXI-DAQ-96-3M : 14 through 65535  
 KPXI-AO-4-1M / KPXI-AO-8-1M: 100 through 16777215

Figure B-4  
Scan timing example

3 Scans, 4 Samples per scan  
(PSC\_Counter=3, NumChan\_Counter=4)



**NOTE** If the card is **Simultaneous AD card** (e.g. KPXI-SDAQ-4-2M), the parameter **SamplIntrv** is of no use.

**SyncMode:** Whether this operation is performed synchronously or asynchronously. If pre-/middle trigger mode is enabled by calling **KDAQ\_AI\_Config()**, this operation should be performed *asynchronously*. Valid values:

SYNCH\_OP: synchronous A/D conversion, that is, the function does not return until the A/D operation completes.

ASYNCH\_OP: asynchronous A/D conversion

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel, ErrorInvalidAdRange, ErrorTransferCountTooLarge, ErrorContIoNotAllowed, ErrorInvalidSampleRate, ErrorOpenFile

## KDAQ\_AI\_ContReadMultiChannels

**Description** This function performs continuous A/D conversions on the specified analog input channels at a rate as close as possible to the rate you specified. This function takes advantage of the Keithley PXI DAQ channel-gain that can be set separately for each channel to perform multi-channel/gain analog input.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AI_ContReadMultiChannels (U16 CardNumber,
    U16 NumChans, U16 *Chans, U16 BufId, U32 ReadScans,
    U32 ScanIntrv, U32 SampIntrv, U16 SyncMode)
```

**Visual Basic**

`KDAQ_AI_ContReadMultiChannels (ByVal CardNumber As Integer, ByVal NumChans As Integer, chans As Integer, ByVal BufId As Integer, ByVal ReadScans As Long, ByVal ScanIntrv As Long, ByVal SampIntrv As Long, ByVal SyncMode As Integer) As Integer`

**Parameters**

**CardNumber:** The card id of the card to perform this operation.

**numChans:** The number of analog input channels in the array *Chans*. Valid values:

- KPXI-SDAQ-4-2M: 1 through 4
- KPXI-SDAQ-4-500K: 1 through 4
- KPXI-DAQ-64-3M: 1 through 512
- KPXI-DAQ-64-500K: 1 through 512
- KPXI-DAQ-64-250K: 1 through 512
- KPXI-DAQ-96-3M: 1 through 1024
- KPXI-AO-4-1M: 1 through 8
- KPXI-AO-8-1M: 1 through 4

**Chans:** Array of analog input channel numbers. Valid values:

- KPXI-SDAQ-4-2M: numbers in *Chans* must be within 0 and 3.
- KPXI-SDAQ-4-500K: numbers in *Chans* must be within 0 and 3.
- KPXI-DAQ-64-3M: numbers in *Chans* must be within 0 and 63.
- KPXI-DAQ-64-500K: numbers in *Chans* must be within 0 and 63.
- KPXI-DAQ-64-250K: numbers in *Chans* must be within 0 and 63.
- KPXI-DAQ-96-3M: numbers in *Chans* must be within 0 and 95.
- KPXI-AO-4-1M: numbers in *Chans* must be within 0 and 7.
- KPXI-AO-8-1M: numbers in *Chans* must be within 0 and 3.

**BufId:** The buffer ID (returned from function `KDAQ_AI_ContBufferSetup`) of the buffer containing the acquired data. The size of the buffer with buffer id of *BufId* must have a length equal to the value of parameter *ReadScans* \* (*number of channels per scan*). If double-buffered mode is enabled, The starting buffer id should be 0. You can ignore this argument. Please refer to [AI data format](#) for the data format in the buffer with *BufId*.

**ReadScans:** If double-buffered mode is disabled, the total number of scans to be performed. For double-buffered acquisition, *ReadScans* is the size (in samples) allocated for each channel in the circular buffer. The valid range of the value is 2 through 16777215. This value must be a multiple of 2.

**NOTE** *If Mcounter is enabled, the ReadScans has to be equal to MidOrDlyScans+MCnt.*

**ScanIntrv:** The length of the scan interval (that is, the counter value between the initiation of each scan sequence). The scan rate will be *TimeBase/ScanIntrv*. The value of *TimeBase* depends on the card type.

If the timer base is from *external*, the valid range of the value is 8 through 16777215. If the timer base is *Internal timer*, the valid range of the value is as follows:

- KPXI-SDAQ-4-2M : 20 through 16777215
- KPXI-SDAQ-4-500K : 80 through 16777215

KPXI-DAQ-64-3M : 14 through 16777215  
 KPXI-DAQ-64-500K : 80 through 16777215  
 KPXI-DAQ-64-250K : 160 through 16777215  
 KPXI-DAQ-96-3M : 14 through 16777215  
 KPXI-AO-4-1M / KPXI-AO-8-1M: 100 through 16777215

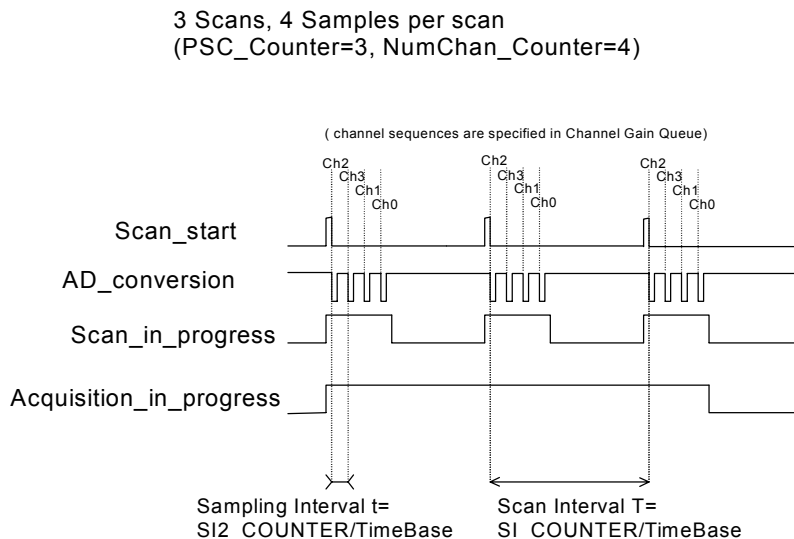
**Samplntrv:** The length of the sample interval (that is, the counter value between each A/D conversion within a scan sequence). The A/D conversion rate will be  $TimeBase/Samplntrv$ . The value of *TimeBase* depends on the card type.

If the timer base is from *external*, the valid range of the value is 8 through 65535.

If the timer base is *Internal timer*, the valid range of the value is as follows:

KPXI-SDAQ-4-2M : invalid  
 KPXI-SDAQ-4-500K : invalid  
 KPXI-DAQ-64-3M : 14 through 65535  
 KPXI-DAQ-64-500K : 80 through 65535  
 KPXI-DAQ-64-250K : 160 through 65535  
 KPXI-DAQ-96-3M : 14 through 65535  
 KPXI-AO-4-1M / KPXI-AO-8-1M: 100 through 16777215

Figure B-5  
**Scan timing example**



**NOTE** If the card is a **Simultaneous AD card** (e.g. KPXI-SDAQ-4-2M), the parameter *Samplntrv* is of no use.

**SyncMode:** Whether this operation is performed synchronously or asynchronously. If pre-/middle trigger mode is enabled by calling **KDAQ\_AI\_Config()**, this operation should be performed **asynchronously**. Valid values:

SYNCH\_OP: synchronous A/D conversion, that is, the function does not return until the A/D operation completes.

ASYNCH\_OP: asynchronous A/D conversion

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel, ErrorInvalidSampleRate, ErrorInvalidAdRange, ErrorTransferCountTooLarge, ErrorContIoNotAllowed

### KDAQ\_AI\_ContReadMultiChannelsToFile

**Description** This function performs continuous A/D conversions on the specified analog input channels at a rate as close as possible to the rate you specified and saves the acquired data in a disk file. The data is written to disk in binary format, with the lower byte first (little endian). Please refer to [DATA file format](#) for the data file structure and [AI data format](#) for the format of the data in the data file. This function takes advantage of the Keithley PXI DAQ channel-gain that can be set separately for each channel to perform multi-channel/gain analog input.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AI_ContReadMultiChannelsToFile (U16 CardNumber,
    U16 NumChans, U16 *Chans, U16 BufId, U8 *FileName,
    U32 ReadScans, U32 ScanIntrv, U32 SampIntrv, U16 SyncMode)
```

#### Visual Basic

```
KDAQ_AI_ContScanChannelsToFile (ByVal wCardNumber As Integer,
    ByVal wChannel As Integer, ByVal BufId As Integer,
    ByVal FileName As String, ByVal ReadScans As Long,
    ByVal ScanIntrv As Long, ByVal SampIntrv As Long,
    ByVal SyncMode As Integer) As Integer
```

**Parameters** **CardNumber:** The card id of the card to perform this operation.

**numChans:** The number of analog input channels in the array *Chans*. Valid values:

- KPXI-SDAQ-4-2M: 1 through 4
- KPXI-SDAQ-4-500K: 1 through 4
- KPXI-DAQ-64-3M: 1 through 512
- KPXI-DAQ-64-500K: 1 through 512
- KPXI-DAQ-64-250K: 1 through 512
- KPXI-DAQ-96-3M: 1 through 1024
- KPXI-AO-4-1M: 1 through 8
- KPXI-AO-8-1M: 1 through 4

**Chans:** Array of analog input channel numbers. Valid values:

- KPXI-SDAQ-4-2M: numbers in *Chans* must be within 0 and 3.
- KPXI-SDAQ-4-500K: numbers in *Chans* must be within 0 and 3.
- KPXI-DAQ-64-3M: numbers in *Chans* must be within 0 and 63.
- KPXI-DAQ-64-500K: numbers in *Chans* must be within 0 and 63.
- KPXI-DAQ-64-250K: numbers in *Chans* must be within 0 and 63.
- KPXI-DAQ-96-3M: numbers in *Chans* must be within 0 and 95.
- KPXI-AO-4-1M: numbers in *Chans* must be within 0 and 7.
- KPXI-AO-8-1M: numbers in *Chans* must be within 0 and 3.

**BufId:** The buffer ID (returned from function *KDAQ\_AI\_ContBufferSetup*) of the buffer containing the acquired data. The size of the buffer with buffer id of

*BufId* must have a length equal to the value of parameter *ReadScans* \* (*number of channels per scan*). If double-buffered mode is enabled, The starting buffer id should be 0. You can ignore this argument. Please refer to [AI data format](#) for the data format in the buffer with *BufId*.

**FileName:** Name of data file which stores the acquired data

**ReadScans:** If double-buffered mode is disabled, the total number of scans to be performed. For double-buffered acquisition, *ReadScans* is the size (in samples) allocated for each channel in the circular buffer. The valid range of the value is 2 through 16777215. This value must be a multiple of 2.

**NOTE** If *Mcounter* is enabled, the *ReadScans* has to be equal to *MidOrDlyScans+MCnt*.

**ScanIntrv:** The length of the scan interval (that is, the counter value between the initiation of each scan sequence). The scan rate will be *TimeBase/ScanIntrv*. The value of *TimeBase* depends on the card type.

If the timer base is from *external*, the valid range of the value is 8 through 16777215. If the timer base is *Internal timer*, the valid range of the value is as follows:

KPXI-SDAQ-4-2M : 20 through 16777215  
KPXI-SDAQ-4-2M : 20 through 16777215  
KPXI-SDAQ-4-500K : 80 through 16777215  
KPXI-DAQ-64-3M : 14 through 16777215  
KPXI-DAQ-64-500K : 80 through 16777215  
KPXI-DAQ-64-250K : 160 through 16777215  
KPXI-DAQ-96-3M : 14 through 16777215  
KPXI-AO-4-1M / KPXI-AO-8-1M: 100 through 16777215

**SamplIntrv:** The length of the sample interval (that is, the counter value between each A/D conversion within a scan sequence). The A/D conversion rate will be *TimeBase/SamplIntrv*. The value of *TimeBase* depends on the card type.

If the timer base is from *external*, the valid range of the value is 8 through 65535.

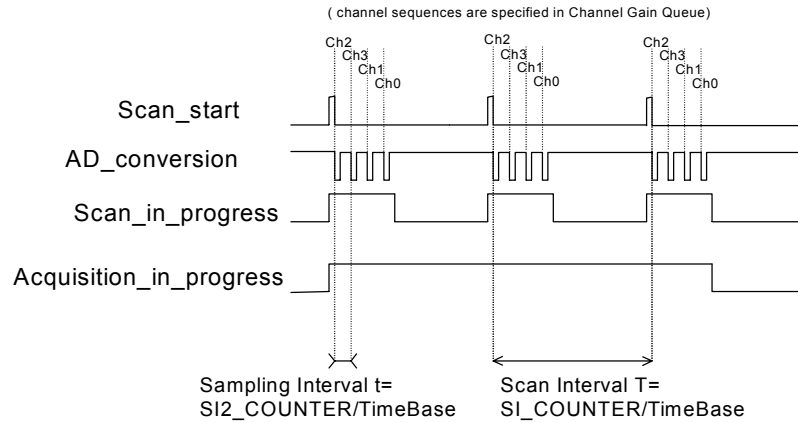
If the timer base is *Internal timer*, the valid range of the value is as follows:

KPXI-SDAQ-4-2M : invalid  
KPXI-SDAQ-4-500K : invalid  
KPXI-DAQ-64-3M : 14 through 65535  
KPXI-DAQ-64-500K : 80 through 65535  
KPXI-DAQ-64-250K : 160 through 65535  
KPXI-DAQ-96-3M : 14 through 65535  
KPXI-AO-4-1M / KPXI-AO-8-1M: 100 through 16777215



Figure B-6  
Scan timing example

3 Scans, 4 Samples per scan  
(PSC\_Counter=3, NumChan\_Counter=4)



**NOTE** If the card is a **Simultaneous AD card** (e.g. KPXI-SDAQ-4-2M), the parameter **SamplIntrv** is invalid.

**SyncMode:** Specifies whether this operation is performed synchronously or asynchronously. If any trigger mode is enabled by calling **KDAQ\_AI\_Config()**, this operation should be performed **asynchronously**. Valid values:

SYNCH\_OP: synchronous A/D conversion, that is, the function does not return until the A/D operation completes.

ASYNCH\_OP: asynchronous A/D conversion

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel, ErrorInvalidSampleRate, ErrorInvalidAdRange, ErrorTransferCountTooLarge, ErrorContIoNotAllowed, ErrorOpenFile

### KDAQ\_AI\_ContScanChannels

**Description** This function performs continuous A/D conversions on the specified continuous analog input channels at a rate as close as possible to the rate you specified. This function takes advantage of the hardware simultaneous or auto-scan functionality to perform multi-channel analog input.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AI_ContScanChannels (U16 CardNumber, U16 Channel,
    U16 BufId, U32 ReadScans, U32 ScanIntrv, U32 SampIntrv,
    U16 SyncMode)
```

## Visual Basic

```
KDAQ_AI_ContScanChannels (ByVal wCardNumber As Integer,
    ByVal wChannel As Integer, ByVal BufId As Integer,
    ByVal ReadScans As Long, ByVal ScanIntrv As Long,
    ByVal SampIntrv As Long, ByVal SyncMode As Integer)
    As Integer
```

### Parameters

**CardNumber:** The card id of the card to perform this operation.

**Channel:** The largest channel number of specified continuous analog input channels. The channel order for acquiring data is as follows:

KPXI-SDAQ-4-2M: number of *Channel* must be within 0 and 3. The continuous scan sequence is ascending and the first one must be zero. For example: 0, 1, 2, 3.

KPXI-SDAQ-4-500K: number of *Channel* must be within 0 and 3. The continuous scan sequence is ascending and the first one must be zero. For example: 0, 1, 2, 3.

KPXI-DAQ-64-3M: number of *Channel* must be within 0 and 63. The continuous scan sequence is ascending and the first one must be zero. For example: 0, 1, 2, 3.

KPXI-DAQ-64-500K: number of *Channel* must be within 0 and 63. The continuous scan sequence is ascending and the first one must be zero. For example: 0, 1, 2, 3.

KPXI-DAQ-64-250K: number of *Channel* must be within 0 and 63. The continuous scan sequence is ascending and the first one must be zero. For example: 0, 1, 2, 3.

KPXI-DAQ-96-3M: number of *Channel* must be within 0 and 95. The continuous scan sequence is ascending and the first one must be zero. For example: 0, 1, 2, 3.

KPXI-AO-4-1M: number of *Channel* must be within 0 and 7. The continuous scan sequence is ascending and the first one must be zero. For example: 0, 1, 2, 3.

KPXI-AO-8-1M: number of *Channel* must be within 0 and 3. The continuous scan sequence is ascending and the first one must be zero. For example: 0, 1, 2, 3.

**BufId:** The buffer ID (returned from function *KDAQ\_AI\_ContBufferSetup*) of the buffer containing the acquired data. The size of the buffer with buffer id of *BufId* must have a length equal to the value of parameter *ReadScans* \* (*number of channels per scan*). If double-buffered mode is enabled, The starting buffer id should be 0. You can ignore this argument. Please refer to [AI data format](#) for the data format in the buffer with *BufId*.

**ReadScans:** If double-buffered mode is disabled, the total number of scans to be performed. For double-buffered acquisition, *ReadScans* is the size (in samples) allocated for each channel in the circular buffer. The valid range of the value is 2 through 16777215. This value must be a multiple of 2.

**NOTE** If *Mcounter* is enabled, the *ReadScans* has to be equal to *MidOrDlyScans+MCnt*.

**ScanIntrv:** The length of the scan interval (that is, the counter value between the initiation of each scan sequence). The scan rate will be *TimeBase/ScanIntrv*. The value of *TimeBase* depends on the card type.

If the timer base is from *external*, the valid range of the value is 8 through 16777215. If the timer base is *Internal timer*, the valid range of the value is as follows:

- KPXI-SDAQ-4-2M : 20 through 16777215
- KPXI-SDAQ-4-500K : 80 through 16777215
- KPXI-DAQ-64-3M : 14 through 16777215
- KPXI-DAQ-64-500K : 80 through 16777215
- KPXI-DAQ-64-250K : 160 through 16777215
- KPXI-DAQ-96-3M : 14 through 16777215
- KPXI-AO-4-1M : 100 through 16777215
- KPXI-AO-8-1M : 100 through 16777215

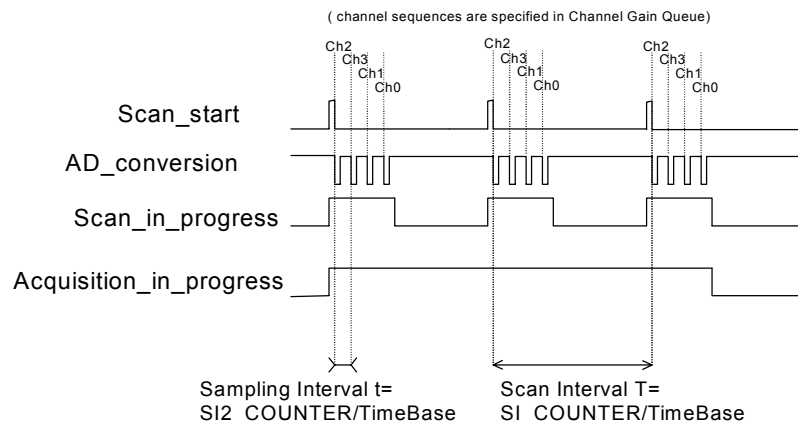
**SamplIntrv:** The length of the sample interval (that is, the counter value between each A/D conversion within a scan sequence). The A/D conversion rate will be *TimeBase/SamplIntrv*. The value of *TimeBase* depends on the card type.

If the timer base is from *external*, the valid range of the value is 8 through 65535. If the timer base is *Internal timer*, the valid range of the value is as follows:

- KPXI-SDAQ-4-2M : invalid
- KPXI-SDAQ-4-500K : invalid
- KPXI-DAQ-64-3M : 14 through 65535
- KPXI-DAQ-64-500K : 80 through 65535
- KPXI-DAQ-64-250K : 160 through 65535
- KPXI-DAQ-96-3M : 14 through 65535
- KPXI-AO-4-1M : 100 through 16777215
- KPXI-AO-8-1M : 100 through 16777215

Figure B-7  
Scan timing example

3 Scans, 4 Samples per scan  
(PSC\_Counter=3, NumChan\_Counter=4)



**NOTE** If the card is a **Simultaneous AD card** (e.g. KPXI-SDAQ-4-2M), the parameter *SamplIntrv* is invalid.

**SyncMode:** Whether this operation is performed synchronously or asynchronously. If pre-/middle trigger mode is enabled by calling `KDAQ_AI_Config()`, this operation should be performed **asynchronously**. Valid values:

SYNCH\_OP: synchronous A/D conversion, that is, the function does not return until the A/D operation completes.

ASYNCH\_OP: asynchronous A/D conversion

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel, ErrorInvalidSampleRate, ErrorInvalidAdRange, ErrorTransferCountTooLarge, ErrorContIoNotAllowed, ErrorLastChannelNotZero, ErrorDiffRangeNotSupport, ErrorChannelNotDescending, ErrorChannelNotAscending

## KDAQ\_AI\_ContScanChannelsToFile

**Description** This function performs continuous A/D conversions on the specified continuous analog input channels at a rate as close as possible to the rate you specified and saves the acquired data in a disk file. The data is written to disk in binary format, with the lower byte first (little endian). Please refer to [DATA file format](#) for the data file structure and [AI data format](#) for the format of the data in the data file. This function takes advantage of the hardware simultaneous or auto-scan functionality to perform multi-channel analog input.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AI_ContScanChannelsToFile (U16 CardNumber,
    U16 Channel, U16 BufId, U8 *FileName, U32 ReadScans,
    U32 ScanIntrv, U32 SampIntrv, U16 SyncMode);
```

### Visual Basic

```
KDAQ_AI_ContScanChannelsToFile (ByVal wCardNumber As Integer,
    ByVal wChannel As Integer, ByVal BufId As Integer,
    ByVal FileName As String, ByVal ReadScans As Long,
    ByVal ScanIntrv As Long, ByVal SampIntrv As Long,
    ByVal SyncMode As Integer) As Integer
```

**Parameters** **CardNumber:** The card id of the card to perform this operation.

**Channel:** The largest channel number of specified continuous analog input channels. The channel order for acquiring data is as follows:

KPXI-SDAQ-4-2M: number of *Channel* must be within 0 and 3. The continuous scan sequence is ascending and the first one must be zero. For example: 0, 1, 2, 3.

KPXI-SDAQ-4-500K: number of *Channel* must be within 0 and 3. The continuous scan sequence is ascending and the first one must be zero. For example: 0, 1, 2, 3.

KPXI-DAQ-64-3M: number of *Channel* must be within 0 and 63. The continuous scan sequence is ascending and the first one must be zero. For example: 0, 1, 2, 3.

KPXI-DAQ-64-500K: number of *Channel* must be within 0 and 63. The continuous scan sequence is ascending and the first one must be zero. For example: 0, 1, 2, 3.

KPXI-DAQ-64-250K: number of *Channel* must be within 0 and 63. The continuous scan sequence is ascending and the first one must be zero. For example: 0, 1, 2, 3.

KPXI-DAQ-96-3M: number of *Channel* must be within 0 and 95. The continuous scan sequence is ascending and the first one must be zero. For example: 0, 1, 2, 3.

KPXI-AO-4-1M: number of *Channel* must be within 0 and 7. The continuous scan sequence is ascending and the first one must be zero. For example: 0, 1, 2, 3.

KPXI-AO-8-1M: number of *Channel* must be within 0 and 3. The continuous scan sequence is ascending and the first one must be zero. For example: 0, 1, 2, 3.

**BufId:** The buffer ID (returned from function *KDAQ\_AI\_ContBufferSetup*) of the buffer containing the acquired data. The size of the buffer with buffer id of *BufId* must have a length equal to the value of parameter *ReadScans* \* (*number of channels per scan*). If double-buffered mode is enabled, The starting buffer id should be 0. You can ignore this argument. Please refer to [AI data format](#) for the data format in the buffer with *BufId*.

**FileName:** Name of data file which stores the acquired data

**ReadScans:** If double-buffered mode is disabled, the total number of scans to be performed. For double-buffered acquisition, *ReadScans* is the size (in samples) allocated for each channel in the circular buffer. The valid range of the value is 2 through 16777215. This value must be a multiple of 2.

**NOTE** If *Mcounter* is enabled, the *ReadScans* has to be equal to *MidOrDlyScans+MCnt*.

**ScanIntrv:** The length of the scan interval (that is, the counter value between the initiation of each scan sequence). The scan rate will be *TimeBase/ScanIntrv*. The value of *TimeBase* depends on the card type.

If the timer base is from *external*, the valid range of the value is 8 through 16777215. If the timer base is *Internal timer*, the valid range of the value is as follows:

- KPXI-SDAQ-4-2M : 20 through 16777215
- KPXI-SDAQ-4-500K : 80 through 16777215
- KPXI-DAQ-64-3M : 14 through 16777215
- KPXI-DAQ-64-500K : 80 through 16777215
- KPXI-DAQ-64-250K : 160 through 16777215

KPXI-DAQ-96-3M : 14 through 16777215  
 KPXI-AO-4-1M / KPXI-AO-8-1M: 100 through 16777215

**Samplntrv:** The length of the sample interval (that is, the counter value between each A/D conversion within a scan sequence). The A/D conversion rate will be  $TimeBase/Samplntrv$ . The value of *TimeBase* depends on the card type.

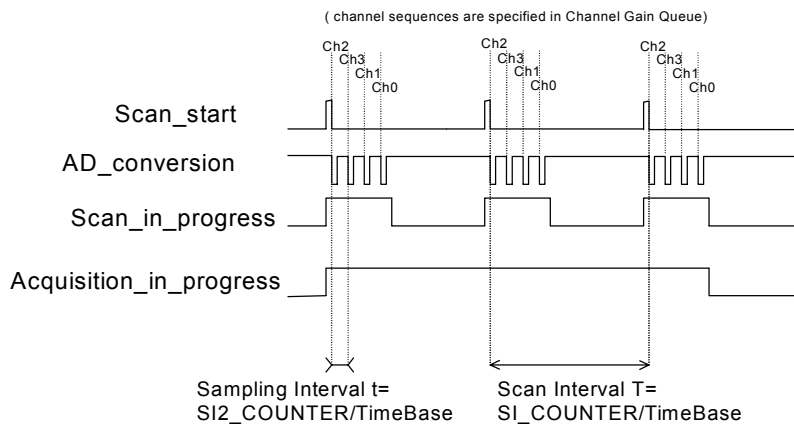
If the timer base is from *external*, the valid range of the value is 8 through 65535.

If the timer base is *Internal timer*, the valid range of the value is as follows:

KPXI-SDAQ-4-2M : invalid  
 KPXI-SDAQ-4-500K : invalid  
 KPXI-DAQ-64-3M : 14 through 65535  
 KPXI-DAQ-64-500K : 80 through 65535  
 KPXI-DAQ-64-250K : 160 through 65535  
 KPXI-DAQ-96-3M : 14 through 65535  
 KPXI-AO-4-1M / KPXI-AO-8-1M: 100 through 16777215

Figure B-8  
**Scan timing example**

3 Scans, 4 Samples per scan  
 (PSC\_Counter=3, NumChan\_Counter=4)



**NOTE** If the card is a **Simultaneous AD card** (e.g. KPXI-SDAQ-4-2M), the parameter **Samplntrv** is invalid.

**SyncMode:** Sets whether this operation is performed synchronously or asynchronously. If pre-/middle trigger mode is enabled by calling **KDAQ\_AI\_Config()**, this operation should be performed **asynchronously**. Valid values:

SYNCH\_OP: synchronous A/D conversion, that is, the function does not return until the A/D operation completes.

ASYNCH\_OP: asynchronous A/D conversion

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel, ErrorInvalidSampleRate,

ErrorInvalidAdRange, ErrorTransferCountTooLarge, ErrorContloNotAllowed, ErrorLastChannelNotZero, ErrorDiffRangeNotSupport, ErrorChannelNotDescending, ErrorChannelNotAscending

### KDAQ\_AI\_ContStatus

**Description** While performing continuous A/D conversions, this function is called to get the A/D status. Please refer to the manual for your device for the AI status the device might meet.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AI_ContStatus (U16 CardNumber, U16 *Status)
```

#### Visual Basic

```
KDAQ_AI_ContStatus (ByVal CardNumber As Integer, Status As Integer) As Integer
```

**Parameters** **CardNumber:** The card id of the card to perform this operation.

**Status:** The continuous AI status returned. The description of the parameter *Status* for various card types is the following:

KPXI-SDAQ-4-2M:  
 KPXI-SDAQ-4-500K:  
 KPXI-DAQ-64-3M:  
 KPXI-DAQ-64-500K:  
 KPXI-DAQ-64-250K:  
 KPXI-DAQ-96-3M:

bit 0 : '1' indicates A/D FIFO is empty  
 bit 1 : '1' indicates A/D FIFO is Half Full  
 bit 2 : '1' indicates A/D FIFO is Full  
 bit 3 : ' ' not used  
 bit 4 : '1' indicates A/D Over Speed Status  
 bit 5 : '1' indicates A/D Overrun Status  
 bit 6 : '1' indicates A/D Trigger Status  
 bit 7 : '1' indicates Scan Counter Terminal Count Status  
 bit 8 through 15 : not used

KPXI-AO-4-1M / KPXI-AO-8-1M:

bit 0 : '1' indicates A/D FIFO is empty  
 bit 1 : '1' indicates A/D FIFO is Half Full  
 bit 2 : '1' indicates A/D FIFO is Full  
 bit 3 : '1' indicates A/D FIFO is Almost Empty  
 bit 4 : '1' indicates A/D FIFO is Almost Full  
 bit 5 : '1' indicates A/D Trigger Status  
 bit 6 : '1' indicates A/D programming Status  
 bit 7 through 15 : not used

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered

## KDAQ\_AI\_ContVScale

**Description** This function converts the values of an array of acquired binary data from a continuous A/D conversion call to the actual input voltages. The acquired binary data in the reading array might include the channel information (please refer to continuous functions, KDAQ\_AI\_ContReadChannel or KDAQ\_AI\_ContScanChannels, for the detailed data format); however, The calculated voltage values in the voltage array returned will not include the channel message.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AI_ContVScale (U16 wCardNumber, U16 adRange,
    void *readingArray, F64 *voltageArray, I32 count)
```

### Visual Basic

```
KDAQ_AI_ContVScale (ByVal CardNumber As Integer,
    ByVal AdRange As Integer, readingArray As Integer,
    voltageArray As Double, ByVal count As Long) As Integer
```

**Parameters** **CardNumber:** The card id of the card to perform this operation.

**AdRange:** The analog input range the continuous specified channel is setting. Please refer to [AI range codes](#) for the valid range values.

**readingArray:** Acquired continuous analog input data array

**voltageArray:** The computed voltages array returned

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidAdRange

## KDAQ\_AI\_DelayTrig\_Config

**Description** Informs KDAQ-DRVR library of the conversion clock source and trigger properties for the Keithley PXI DAQ device that performs delay triggered data acquisition operation.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AI_DelayTrig_Config (U16 wCardNumber, U16 ClkSrc,
    U32 TrigSrcCtrl, U32 DlyScans, U16 ReTrgEn, U16 ReTrgCnt,
    BOOLEAN AutoResetBuf)
```

### Visual Basic

```
KDAQ_AI_DelayTrig_Config (ByVal CardNumber As Integer,
    ByVal ClkSrc As Integer, ByVal TrigSrcCtrl As Integer,
    ByVal DlyScans As Long, ByVal ReTrgEn As Integer,
    ByVal ReTrgCnt As Integer, ByVal AutoResetBuf As Byte)
    As Integer
```



**Parameters**

**CardNumber:** The card id of the card to perform this operation.

**ClkSrc:** The setting for A/D clock sources. This argument is an integer expression formed from one or more of the manifest constants defined in Kdaqdrv.h. There are two groups of constants:

**(1) A/D Conversion Source Selection**

KDAQ\_AI\_ADCONVSRC\_Int : Internal timer (default)

KDAQ\_AI\_ADCONVSRC\_AFI0: From AFI0 pin

KDAQ\_AI\_ADCONVSRC\_SSI: From SSI source

KDAQ\_AI\_ADCONVSRC\_AFI1: From AFI1 pin (*only available for KPXI-AO-8-1M and KPXI-AO-4-1M*)

**(2) A/D Delay Counter Source Selection (*only available for KPXI-AO-8-1M and KPXI-AO-4-1M*)**

KDAQ\_AI\_DTSRC\_Int : Internal timer (default)

KDAQ\_AI\_DTSRC\_AFI1: From AFI1 pin

KDAQ\_AI\_DTSRC\_GPTC0: From GPTC0\_OUT

KDAQ\_AI\_DTSRC\_GPTC1: From GPTC1\_OUT

When two or more constants are used to form the *ClkSrc* argument, the constants are combined with the bitwise-OR operator(`|`).

**TrigSrcCtrl:** The setting for A/D Trigger source control. This argument is an integer expression formed from one or more of the manifest constants defined in kdaqdrv.h. There are three groups of constants:

**(1) Trigger Source Selection**

KDAQ\_AI\_TRGSRC\_SOFT: software (default)

KDAQ\_AI\_TRGSRC\_ANA: From analog trigger pin

KDAQ\_AI\_TRGSRC\_ExtD: From external digital trigger pin

KDAQ\_AI\_TRSRC\_SSI: From SSI source

**(2) Delay Source Selection**

KDAQ\_AI\_Dly1nSamples: delay in samples

KDAQ\_AI\_Dly1nTimebase: delay in time base (default)

**(3) External Digital Trigger Polarity**

KDAQ\_AI\_TrgPositive: Trigger positive edge active (default)

KDAQ\_AI\_TrgNegative: Trigger negative edge active

When two or more constants are used to form the *TrigSrcCtrl* argument, the constants are combined with the bitwise-OR operator(`|`).

***DlyScans***: The number of data or timer ticks will be ignored after a specific trigger event. The valid value range of *DlyScans* is 0 through 65535.

***ReTrgEn***: 0: Re-trigger in an acquisition is disabled. (default value)

1: Re-trigger in an acquisition is enabled.

***ReTrgCnt***: The accepted trigger times in an acquisition. If the value of *ReTrgCnt* is 0, the AI operation will be triggered **infinitely**. The valid value range of *ReTrgCnt* is 0 through 65535.

**NOTE** *To enable infinite re-trigger mode of continuous AI, Calling KDAQ\_AI\_DelayTrig\_Config with '1' value of ReTrgEn and zero value of ReTrgCnt.*

***AutoResetBuf***:

FALSE: The AI buffers set by function *KDAQ\_AI\_ContBufferSetup* are retained and must call the function *KDAQ\_AI\_ContBufferReset* to reset the buffer

TRUE: The AI buffers set by function *KDAQ\_AI\_ContBufferSetup* are reset automatically by the driver while the AI operation is finished

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport

## KDAQ\_AI\_EventCallback

**Description** Controls and notifies the user's application when a specified DAQ event occurs. The notification is performed through a user-specified callback function.

For windows version, the event message will be removed automatically after calling *KDAQ\_AI\_Async\_Clear*. The event message can also be manually removed by set the parameter "*mode*" to be 0.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AI_EventCallback (U16 CardNumber, I16 mode,
    I16 EventType, U32 callbackAddr)
```

**Visual Basic 5**

```
KDAQ_AI_EventCallback (ByVal CardNumber As Integer,
    ByVal mode As Integer, ByVal EventType As Integer,
    ByVal callbackAddr As Long) As Integer
```

**Parameters**    **CardNumber:** The card id of the card to perform this operation.  
**mode:** add or remove the event message. The valid values: 0: remove, 1: add  
**EventType:** event criteria. The valid values are:  
 DAQEnd: Notification for the completeness of asynchronous analog input operation  
 DBEvent: Notification for the next half buffer of data in circular buffer is ready for transfer  
 TrigEvent: Notification for the data associated to the next trigger signal is available  
**callbackAddr:** the address of the user callback function. KDAQ-DRVR calls this function when the specified event occurs. If you wish to remove the event message, set *callbackAddr* to 0.

**Return Value**    NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport

### KDAQ\_AI\_InitialMemoryAllocated

**Description**    This function returns the available memory size for analog input in the device driver in argument *MemSize*. The continuous analog input transfer size can not exceed this size.  
 This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax**        **Microsoft C/C++ and Borland C++**  

```
I16 KDAQ_AI_InitialMemoryAllocated (U16 CardNumber,
                                     U32 *MemSize)
```

**Visual Basic**  

```
KDAQ_AI_InitialMemoryAllocated (ByVal CardNumber As Integer,
                                  MemSize As Long) As Integer
```

**Parameters**    **CardNumber:** The card id of the card to perform this operation.  
**MemSize:** The available memory size for continuous AI in device driver of this card. The unit is KB (1024 bytes).

**Return Value**    NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered

### KDAQ\_AI\_MiddleTrig\_Config

**Description**    Informs KDAQ-DRVR library of the conversion clock source and trigger properties for the Keithley PXI DAQ device that performs middle triggered data acquisition operation.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M

**Syntax****Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AI_MiddleTrig_Config (U16 wCardNumber, U16 ClkSrc,
    U32 TrigSrcCtrl, U32 MiddleScans, U16 MCtrEn, U16 MCnt,
    BOOLEAN AutoResetBuf)
```

**Visual Basic**

```
KDAQ_AI_MiddleTrig_Config (ByVal CardNumber As Integer,
    ByVal ClkSrc As Integer, ByVal TrigSrcCtrl As Integer,
    ByVal MiddleScans As Long, ByVal MCtrEn As Integer,
    ByVal MCnt As Integer, ByVal AutoResetBuf As Byte)
    As Integer
```

**Parameters**

**CardNumber:** The card id of the card to perform this operation.

**ClkSrc:** The setting for A/D clock source. This argument is an integer expression formed from one or more of the manifest constants defined in Kdaqdrv.h. There is one group of constants:

**(1) A/D Conversion Source Selection**

KDAQ\_AI\_ADCONVSRC\_Int : Internal timer (default)

KDAQ\_AI\_ADCONVSRC\_AFI0: From AFI0 pin

KDAQ\_AI\_ADCONVSRC\_SSI: From SSI source

**TrigSrcCtrl:** The setting for A/D Trigger source control. This argument is an integer expression formed from one or more of the manifest constants defined in kdaqdrv.h. There are two groups of constants:

**(1) Trigger Source Selection**

KDAQ\_AI\_TRGSRC\_SOFT : software (default)

KDAQ\_AI\_TRGSRC\_ANA : From analog trigger pin

KDAQ\_AI\_TRGSRC\_ExtD: From external digital trigger pin

KDAQ\_AI\_TRSRC\_SSI : From SSI source

**(2) External Digital Trigger Polarity**

KDAQ\_AI\_TrgPositive: Trigger positive edge active (default)

KDAQ\_AI\_TrgNegative: Trigger negative edge active

When two or more constants are used to form the *TrigSrcCtrl* argument, the constants are combined with the bitwise-OR operator(|).

**MiddleScans:** The number of data will be accessed after a specific trigger event. The valid value range of *MiddleScans* is 0 through 16777215.

**MCtrEn:** 0: Mcounter is disabled. (default)

1: Mcounter is enabled and then the trigger signal is ignored before M terminal count is reached.

**MCnt:** The counter value of Mcounter. The valid value range of *MCnt* is 0 through 65535.

**AutoResetBuf:**

FALSE: The AI buffers set by function “KDAQ\_AI\_ContBufferSetup” are retained and must call the function “KDAQ\_AI\_ContBufferReset” to reset the buffer

TRUE: The AI buffers set by function “KDAQ\_AI\_ContBufferSetup” are reset automatically by the driver when the AI operation is finished

**NOTE** *If Mcounter is enabled, the ReadScans parameter of continuous AI functions KDAQ\_AI\_ContXXXX has to be equal to MiddleScans+MCnt.*

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport

### KDAQ\_AI\_MuxScanSetup

**Description** Stores *numChans*, *chans*, and *gain\_refGnd* in the Channel-Gain Queue for a scanned data acquisition operation. The function uses this memory table during scanning operations (*KDAQ\_AI\_ReadMuxScan*, *KDAQ\_AI\_ContMuxScan* or *KDAQ\_AI\_ContMuxScanToFile*) to automatically sequence through an arbitrary set of analog input channels and to allow gains to automatically change during scanning. This function is only available for **Multiplexed AD card** (e.g. KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, except KPXI-AO-8-1M and KPXI-AO-4-1-M). If any of AI functions other than *KDAQ\_AI\_ReadMuxScan*, *KDAQ\_AI\_ContMuxScan* or *KDAQ\_AI\_ContMuxScanToFile* are called, the channel-gain queue will be modified.

This function is supported by the following models: KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AI_MuxScanSetup (U16 wCardNumber, U16 wNumChans,
    U16* Chans, U16* AdRange_RefGnds);
```

**Visual Basic**

```
KDAQ_AI_MuxScanSetup (ByVal CardNumber As Integer,
    ByVal wNumChans As Integer, chans As Integer,
    AdRange_RefGnds As Integer) As Integer
```

**Parameters** **CardNumber:** The card id of the card to perform this operation.

**numChans:** The number of analog input channels in the array *Chans*. Valid values:

KPXI-DAQ-64-3M: 1 through 512  
 KPXI-DAQ-64-500K: 1 through 512

KPXI-DAQ-64-250K: 1 through 512

KPXI-DAQ-96-3M: 1 through 1024

**Chans:** Array of analog input channel numbers.

KPXI-DAQ-64-3M: numbers in *Chans* must be within 0 and 63.

KPXI-DAQ-64-500K: numbers in *Chans* must be within 0 and 63.

KPXI-DAQ-64-250K: numbers in *Chans* must be within 0 and 63.

KPXI-DAQ-96-3M: numbers in *Chans* must be within 0 and 95.

**AdRange\_RefGnds** : An integer array of length numChans that contains the analog input range and reference ground for every channel in array Chans. Please refer to KDAQ\_AI\_CH\_Config section for the setting of *AdRange\_RefGnd* for each channel.

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel, ErrorInvalidAdRange

## KDAQ\_AI\_PostTrig\_Config

**Description** Informs KDAQ-DRVR library of the conversion clock source and trigger properties for the Keithley PXI DAQ device to perform post triggered data acquisition operation.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AI_PostTrig_Config (U16 wCardNumber, U16 ClkSrc,
    U32 TrigSrcCtrl, U16 ReTrgEn, U16 ReTrgCnt,
    BOOLEAN AutoResetBuf)
```

### Visual Basic

```
KDAQ_AI_PostTrig_Config (ByVal CardNumber As Integer,
    ByVal ClkSrc As Integer, ByVal TrigSrcCtrl As Integer,
    ByVal ReTrgEn As Integer, ByVal ReTrgCnt As Integer,
    ByVal AutoResetBuf As Byte) As Integer
```

**Parameters** **CardNumber:** The card id of the card to perform this operation.

**ClkSrc:** The setting for A/D clock sources. This argument is an integer expression formed from one or more of the manifest constants defined in Kdaqdrv.h. There is one group of constants:

### (1) A/D Conversion Source Selection

KDAQ\_AI\_ADCONVSRC\_Int : Internal timer (default)

KDAQ\_AI\_ADCONVSRC\_AFI0: From AFI0 pin

KDAQ\_AI\_ADCONVSRC\_SSI: From SSI source

KDAQ\_AI\_ADCONVSRC\_AFI1: From AFI1 pin (**only available for KPXI-AO-4-1M and KPXI-AO-8-1M**)

**TrigSrcCtrl:** The setting for A/D Trigger source control. This argument is an integer expression formed from one or more of the manifest constants defined in kdaqdrvr.h. There are two groups of constants:

**(1) Trigger Source Selection**

KDAQ\_AI\_TRGSRC\_SOFT : software (default)

KDAQ\_AI\_TRGSRC\_ANA : From analog trigger pin

KDAQ\_AI\_TRGSRC\_ExtD: From external digital trigger pin

KDAQ\_AI\_TRSRC\_SSI : From SSI source

**(2) External Digital Trigger Polarity**

KDAQ\_AI\_TrgPositive: Trigger positive edge active (default)

KDAQ\_AI\_TrgNegative: Trigger negative edge active

When two or more constants are used to form the *TrigSrcCtrl* argument, the constants are combined with the bitwise-OR operator(|).

**ReTrgEn:** 0: Re-trigger in an acquisition is disabled (default value), 1: Re-trigger in an acquisition is enabled.

**ReTrgCnt:** The accepted trigger times in an acquisition. If the value of ReTrgCnt is 0, the AI operation will be triggered **infinitely**. The valid value range of *ReTrgCnt* is 0 through 65535.

**NOTE** *To cause an infinite re-trigger mode of continuous AI, Calling KDAQ\_AI\_PostTrig\_Config with '1' value of ReTrgEn and zero value of ReTrgCnt.*

**AutoResetBuf.**

FALSE: The AI buffers set by function “KDAQ\_AI\_ContBufferSetup” are retained and must call the function “KDAQ\_AI\_ContBufferReset” to reset the buffer

TRUE: The AI buffers set by function “KDAQ\_AI\_ContBufferSetup” are reset automatically by the driver while the AI operation is finishing

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport

**KDAQ\_AI\_PreTrig\_Config**

**Description** Informs KDAQ-DRVR library of the conversion clock source and trigger properties for the Keithley PXI DAQ device that performs pre-triggered data acquisition operation.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M

**Syntax****Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AI_PreTrig_Config (U16 wCardNumber, U16 ClkSrc,
    U32 TrigSrcCtrl, U16 MCtrEn, U16 MCnt,
    BOOLEAN AutoResetBuf)
```

**Visual Basic**

```
KDAQ_AI_PreTrig_Config (ByVal CardNumber As Integer,
    ByVal ClkSrc As Integer, ByVal TrigSrcCtrl As Integer,
    ByVal MCtrEn As Integer, ByVal MCnt As Integer,
    ByVal AutoResetBuf As Byte) As Integer
```

**Parameters**

**CardNumber:** The card id of the card to perform this operation.

**ClkSrc:** The setting for A/D clock source. This argument is an integer expression formed from one or more of the manifest constants defined in Kdaqdrv.h. There is one group of constants:

**(1) A/D Conversion Source Selection**

KDAQ\_AI\_ADCONVSRC\_Int : Internal timer (default)

KDAQ\_AI\_ADCONVSRC\_AFI0: From AFI0 pin

KDAQ\_AI\_ADCONVSRC\_SSI: From SSI source

**TrigSrcCtrl:** The setting for A/D Trigger source control. This argument is an integer expression formed from one or more of the manifest constants defined in kdaqdrv.h. There are two groups of constants:

**(1) Trigger Source Selection**

KDAQ\_AI\_TRGSRC\_SOFT : software (default)

KDAQ\_AI\_TRGSRC\_ANA : From analog trigger pin

KDAQ\_AI\_TRGSRC\_ExtD: From external digital trigger pin

KDAQ\_AI\_TRSRC\_SSI : From SSI source

**(2) External Digital Trigger Polarity**

KDAQ\_AI\_TrgPositive: Trigger positive edge active (default)

KDAQ\_AI\_TrgNegative: Trigger negative edge active



When two or more constants are used to form the *TrigSrcCtrl* argument, the constants are combined with the bitwise-OR operator(`|`).

**MCtrEn:** 0: Mcounter is disabled. (default)

1: Mcounter is enabled and then the trigger signal is ignored before M terminal count is reached

**MCnt:** The counter value of Mcounter. The valid value range of *MCnt* is 0 through 65535.

**AutoResetBuf:**

FALSE: The AI buffers set by function “KDAQ\_AI\_ContBufferSetup” are retained and must call the function “KDAQ\_AI\_ContBufferReset” to reset the buffer

TRUE: The AI buffers set by function “KDAQ\_AI\_ContBufferSetup” are reset automatically by the driver when the AI operation is finishing

**NOTE** *If Mcounter is enabled, the ReadScans parameter of continuous AI functions KDAQ\_AI\_ContXXXX has to be equal to MCnt.*

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport

### KDAQ\_AI\_ReadChannel

**Description** This function performs a software triggered A/D conversion (analog input) on an analog input channel and returns the value converted.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AI_ReadChannel (U16 CardNumber, U16 Channel,
    U16 *Value)
```

**Visual Basic**

```
KDAQ_AI_ReadChannel (ByVal CardNumber As Integer,
    ByVal Channel As Integer, Value As Integer) As Integer
```

**Parameters** **CardNumber:** The card id of the card to perform this operation.

**Channel:** Analog input channel number.

Range: 0 through 3 for KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-AO-8-1M

Range: 0 through 7 for KPXI-AO-4-1M

Range: 0 through 63 for KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K

Range: 0 through 95 for KPXI-DAQ-96-3M

**Value:** The A/D converted value. For the data format in *value*, please refer to the description of *Buffer* argument of `KDAQ_AI_ContReadChannel ()` for the correct data format.

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel, ErrorInvalidAdRange

## KDAQ\_AI\_ReadMuxScan

**Description** Returns readings for all analog input channels selected by *KDAQ\_AI\_MuxScanSetup*. This function is only available for **Multiplexed AD card** (e.g. KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, except KPXI-AO-8-1M and KPXI-AO-4-1M).

This function is supported by the following models: KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AI_ReadMuxScan (U16 wCardNumber, U16 *pwBuffer)
```

### Visual Basic

```
KDAQ_AI_ReadMuxScan (ByVal CardNumber As Integer, Buffer As Integer) As Integer
```

**Parameters** **CardNumber:** The card id of the card to perform this operation.

**Buffer:** An integer array to contain the acquired data. Please refer to [AI data format](#) for the data format in the Buffer.

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel, ErrorInvalidAdRange

## KDAQ\_AI\_ScanReadChannels

**Description** This function performs software triggered A/D conversions (analog input) on analog input channels and returns the values converted. This function is only available for **Multiplexed AD card** (e.g. KPXI-DAQ-64-500K).

This function is supported by the following models: KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AI_ScanReadChannels (U16 wCardNumber, U16 wNumChans, U16 *Chans, U16 *Buffer)
```

### Visual Basic

```
KDAQ_AI_ScanReadChannels (ByVal CardNumber As Integer, ByVal NumChans As Integer, chans As Integer, Buffer As Integer) As Integer
```

**Parameters** **CardNumber:** The card id of the card to perform this operation.

**numChans:** The number of analog input channels in the array *Chans*. The valid value:

KPXI-DAQ-64-3M: 1 through 512  
 KPXI-DAQ-64-500K: 1 through 512  
 KPXI-DAQ-64-250K: 1 through 512  
 KPXI-DAQ-96-3M: 1 through 1024

KPXI-AO-4-1M: 1 through 8  
 KPXI-AO-8-1M: 1 through 4

**Chans:** Array of analog input channel numbers.

KPXI-DAQ-64-3M: numbers in *Chans* must be within 0 and 63.  
 KPXI-DAQ-64-500K: numbers in *Chans* must be within 0 and 63.  
 KPXI-DAQ-64-250K: numbers in *Chans* must be within 0 and 63.  
 KPXI-DAQ-96-3M: numbers in *Chans* must be within 0 and 95.  
 KPXI-AO-4-1M: numbers in *Chans* must be within 0 and 7.  
 KPXI-AO-8-1M: numbers in *Chans* must be within 0 and 3.

**Buffer:** An integer array to contain the acquired data. The length (in samples) of *Buffer* must be equal to or greater the value of parameter *numChans*. Please refer to [AI data format](#) for the data format in the Buffer.

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel, ErrorInvalidAdRange

### KDAQ\_AI\_SimuReadChannel

**Description** This function performs a software triggered A/D conversion (analog input) on analog input channels and returns the values converted. This function is only available for **Simultaneous AD card** (e.g. KPXI-SDAQ-4-2M).

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AI_SimuReadChannel (U16 wCardNumber, U16 wNumChans,
    U16 *Chans, U16 *Buffer)
```

**Visual Basic**

```
KDAQ_AI_SimuReadChannel (ByVal CardNumber As Integer,
    ByVal NumChans As Integer, chans As Integer,
    Buffer As Integer) As Integer
```

**Parameters** **CardNumber:** The card id of the card to perform this operation.

**numChans:** The number of analog input channels in the array *Chans*. The valid value:

KPXI-SDAQ-4-2M: 1 through 4  
 KPXI-SDAQ-4-500K: 1 through 4

**Chans:** Array of analog input channel numbers.

KPXI-SDAQ-4-2M: numbers in *Chans* must be within 0 and 3.  
 KPXI-SDAQ-4-500K: numbers in *Chans* must be within 0 and 3.

**Buffer:** An integer array to contain the acquired data. The length (in samples) of *Buffer* must be equal to or greater the value of parameter *numChans*. Please refer to [AI data format](#) for the data format in the Buffer.

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel, ErrorInvalidAdRange

## KDAQ\_AI\_VoltScale

**Description** This function converts the result from a KDAQ\_AI\_ReadChannel call to the actual input voltage.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AI_VoltScale (U16 CardNumber, U16 AdRange,  
    I16 reading, F64 *voltage)
```

### Visual Basic

```
KDAQ_AI_VoltScale (ByVal CardNumber As Integer,  
    ByVal AdRange As Integer, ByVal reading As Integer,  
    Voltage As Double) As Integer
```

**Parameters** **CardNumber**: The card id of the card to perform this operation.

**AdRange**: The analog input range the specified channel is setting. Please refer to [AI range codes](#) for the valid range values.

**reading**: The result of the AD Conversion.

**voltage**: computed voltage value

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidAdRange

## KDAQ\_AI\_VReadChannel

**Description** This function performs a software triggered A/D conversion (analog input) on an analog input channel and returns the value scaled to a voltage in units of volts.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AI_VReadChannel (U16 CardNumber, U16 Channel,  
    F64 *voltage)
```

### Visual Basic

```
KDAQ_AI_VReadChannel (ByVal CardNumber As Integer,  
    ByVal Channel As Integer, Voltage As Double) As Integer
```

**Parameters**    **CardNumber:** The card id of the card to perform this operation.  
**Channel:** Analog input channel number.  
 Range: 0 through 3 (for KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-AO-8-1M)  
 Range: 0 through 7 (for KPXI-AO-4-1M)  
 Range: 0 through 63 (for KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K)  
 Range: 0 through 95 (for KPXI-DAQ-96-3M)  
**voltage:** The measured voltage value returned and scaled to units of voltage.

**Return Value**    NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel, ErrorInvalidAdRange

**KDAQ\_AIO\_Config**

**Description**    Informs KDAQ-DRVR library of the *Timer Source* and the *Analog Trigger setting* for the Keithley PXI DAQ device with card ID *CardNumber*. You must call this function if using the **external timer source** or the **Analog trigger** mode of AI/AO.  
 This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax**            **Microsoft C/C++ and Borland C++**  
 I16 KDAQ\_AIO\_Config (U16 wCardNumber, U16 TimerBase, U16 AnaTrigCtrl, U16 H\_TrgLevel, U16 L\_TrgLevel)

**Visual Basic**

KDAQ\_AIO\_Config (ByVal CardNumber As Integer, ByVal TimerBase As Integer, ByVal AnaTrigCtrl As Integer, ByVal H\_TrgLevel As Integer, ByVal L\_TrgLevel As Integer) As Integer

**Parameters**    **CardNumber:** The card id of the card to perform this operation.  
**TimerBase:** The Time Base the device selected. Valid values:  
 KDAQ\_IntTimeBase: Internal timer as the time base  
 KDAQ\_ExtTimeBase: External timer as the time base  
 KDAQ\_SSITimeBase: The timer based on the SSI source

**AnaTrigCtrl:** The setting for Analog Trigger control. This argument is an integer expression formed from one or more of the manifest constants defined in kdaqdrv.h. There are seven groups of constants:

**(1) Trigger Source Selection**  
**KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K:**

CH0ATRIG : AI channel 0  
 CH1ATRIG : AI channel 1  
 CH2ATRIG : AI channel 2

CH3ATRIG : AI channel 3  
EXTATRIG : From external analog trigger pin

**KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K,  
KPXI-DAQ-96-3M, KPXI-AO-4-1M, KPXI-AO-8-1M**

ADCATRIG: The first AI channel in the channel-gain queue  
EXTATRIG : From external analog trigger pin

## (2) Trigger Condition Selection

Below\_Low\_level: Below-Low-Level Triggering  
Above\_High\_Level: Above-High-Level Triggering  
Inside\_Region: Inside Region Triggering  
High\_Hysteresis: High Hysteresis Triggering  
Low\_Hysteresis: Low Hysteresis Triggering

When two or more constants are used to form the *AnaTrigCtrl* argument, the constants are combined with the bitwise-OR operator(`|`).

**H\_TrgLevel:** The High value setting of Trigger level. The valid range of the value is 1 through 256. Please refer to the hardware manual for the relationship between the value of *TrgLevel* and trigger voltage.

**L\_TrgLevel:** The Low value setting of Trigger level. The valid range of the value is 1 through 255. Please refer to the hardware manual for the relationship between the value of *TrgLevel* and trigger voltage.

For example: If the trigger voltage is  $\pm 10V$ , the relationship between the value of *TrgLevel* and trigger voltage is contained in [Table B-2](#):

Table B-2

### Example trigger condition selection (KDAQ\_AIO\_Config)

Trigger Level digital setting	Trigger voltage
0xFF	9.92V
0xFE	9.84V
---	---
0x81	0.08V
0x80	0
0x7F	-0.08V
---	---
0x02	-9.92V
0x01	-10V

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport

## KDAQ\_AO\_AsyncCheck

**Description** Check the current status of the asynchronous analog output operation. This function is only available for the device that uses timer pacer (*KDAQ\_DA\_WRSRC\_Int*) as the D/A R/W Source.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K

**Syntax**      **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AO_AsyncCheck (U16 CardNumber, BOOLEAN *Stopped,
    U32 WriteCnt)
```

**Visual Basic**

```
KDAQ_AO_AsyncCheck (ByVal CardNumber As Integer,
    Stopped As Byte, WriteCnt As Long) As Integer
```

**Parameters**      **CardNumber:** The card id of the card that performs the asynchronous operation.

**Stopped:** Whether the asynchronous analog output operation has completed. If *Stopped* = TRUE, the analog output operation has stopped. Either the number of D/A conversions indicated in the call that initiated the asynchronous analog output operation has completed or an error has occurred. If *Stopped* = FALSE, the operation is not yet complete. (constants TRUE and FALSE are defined in kdaqdrv.h)

**WriteCnt:** The number of analog output data that have been written at the time of calling KDAQ\_AO\_AsyncCheck().

**Return Value**      NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport

**KDAQ\_AO\_AsyncClear**

**Description**      Stop the asynchronous analog output operation.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K

**Syntax**      **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AO_AsyncClear (U16 CardNumber, U32 *UpdateCnt,
    U16 stop_mode)
```

**Visual Basic**

```
KDAQ_AO_AsyncClear (ByVal CardNumber As Integer, UpdateCnt As Long,
    stop_mode As Integer) As Integer
```

**Parameters**      **CardNumber:** The card id of the card that performs the asynchronous operation.

**WriteCnt:** The number of analog output data that have been written at the time of calling KDAQ\_AO\_AsyncClear().

**stop\_mode:** The DA transfer termination mode selected. Valid values:

KDAQ\_DA\_TerminateImmediate : Software terminate the DA continuous operation immediately

KDAQ\_DA\_TerminateUC: Software terminate the DA continuous operation on next update counter terminal count

KDAQ\_DA\_TerminateIC: Software terminate the DA continuous operation on iteration count

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport

### KDAQ\_AO\_AsyncDbIBufferHalfReady

**Description** Checks whether the next half buffer is ready for new data during an asynchronous double-buffered analog output operation.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AO_AsyncDbIBufferHalfReady (U16 CardNumber,
    BOOLEAN *HalfReady)
```

#### Visual Basic

```
KDAQ_AI_AsyncDbIBufferHalfReady (ByVal CardNumber As Integer,
    HalfReady As Byte) As Integer
```

**Parameters** **CardNumber:** The card id of the card that performs the asynchronous double-buffered operation.

**HalfReady:** Whether the next half buffer is ready for new data.

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport

### KDAQ\_AO\_AsyncDbIBufferMode

**Description** Enables or disables double-buffered data acquisition mode.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AO_AsyncDbIBufferMode (U16 CardNumber,
    BOOLEAN Enable)
```

#### Visual Basic

```
KDAQ_AO_AsyncDbIBufferMode (ByVal CardNumber As Integer,
    ByVal Enable As Byte) As Integer
```

**Parameters** **CardNumber:** The card id of the card that double-buffered mode is to be set on.

**Enable:** Whether the double-buffered mode is enabled or not.

TRUE: double-buffered mode is enabled.

FALSE: double-buffered mode is disabled. (constants TRUE and FALSE are defined in kdaqdrv.h)

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport

### KDAQ\_AO\_CH\_Config

**Description** Informs KDAQ-DRVR library of the reference voltage value selected for an analog output channel of Keithley PXI DAQ Device. You can configure each channel to use an internal reference of 10V (default) or an external reference (-10V ~ +10V).



After the function "KDAQ\_Register\_Card" is called, all of the analog output channels are configured as bipolar and internal reference source by default. If you wish to perform the device with the default settings, it is not necessary to call this function to configure the channel(s) again. Otherwise, this function has to be called to program the device for the settings you want before calling a function to perform a voltage output operation.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax**

**Microsoft C/C++ and Borland C++**

I16 KDAQ\_AO\_CH\_Config (U16 wCardNumber, U16 wChannel, U16 wOutputPolarity, U16 wIntOrExtRef, F64 refVoltage)

**Visual Basic**

KDAQ\_AO\_CH\_Config (ByVal CardNumber As Integer, ByVal Channel As Integer, ByVal OutputPolarity As Integer, ByVal wIntOrExtRef As Integer, ByVal refVoltage As Double) As Integer

**Parameters**

**CardNumber:** The card id of the card to perform this operation.

**Channel:** The AO channel number configured.

- KPXI-SDAQ-4-2M : 0 through 1 or All\_Channels (-1)
- KPXI-SDAQ-4-500K : 0 through 1 or All\_Channels (-1)
- KPXI-DAQ-64-3M : 0 through 1 or All\_Channels (-1)
- KPXI-DAQ-64-500K : 0 through 1 or All\_Channels (-1)
- KPXI-DAQ-64-250K : 0 through 1 or All\_Channels (-1)
- KPXI-AO-4-1M : 0 through 3 or All\_Channels (-1)
- KPXI-AO-8-1M : 0 through 7 or All\_Channels (-1)

**OutputPolarity:** The polarity (unipolar or bipolar) of the output channel. Valid values:

- KDAQ\_DA\_BiPolar
- KDAQ\_DA\_UniPolar

**IntOrExtref:** The DA reference voltage source of the output channel. Valid values:

- KDAQ\_DA\_Int\_REF : internal reference
- KDAQ\_DA\_Ext\_REF : external reference

**refVoltage:** Voltage reference value.

If the D/A reference voltage source your device use is internal reference, the valid values for *refVoltage* is 10.

If the D/A reference voltage source your device use is external reference, the valid range for *refVoltage* is -10 to +10.

**Return Value**

NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidDaRefVoltage

## KDAQ\_AO\_Config

**Description** Informs KDAQ-DRVR library of the trigger source selected for the Keithley PXI DAQ Device with card ID *CardNumber*. After the function "KDAQ\_Register\_Card" is called, the device is configured as the following by default:

D/A R/W source: KDAQ\_DA\_WRSRC\_Int

D/A trigger mode : KDAQ\_DA\_TRGMOD\_POST

D/A trigger source : KDAQ\_DA\_TRGSRC\_SOFT

Auto reset buffer: Enabled (AutoResetBuf : TRUE)

If you wish to perform the device with the default settings, it is not necessary to call this function to make the configuration again. Otherwise, this function has to be called before calling a function to perform the continuous analog output operation.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AO_Config (U16 wCardNumber, U16 ConfigCtrl,
    U16 TrigCtrl, U16 ReTrgCnt, U16 DLY1Cnt, U16 DLY2Cnt,
    BOOLEAN AutoResetBuf)
```

### Visual Basic

```
KDAQ_AO_Config (ByVal CardNumber As Integer,
    ByVal ConfigCtrl As Integer, ByVal TrigCtrl As Integer,
    ByVal ReTrgCnt As Integer, ByVal DLY1Cnt As Integer,
    ByVal DLY2Cnt As Integer, ByVal AutoResetBuf As Byte)
    As Integer
```

**Parameters** **CardNumber**: The card id of the card to perform this operation.

**ConfigCtrl**: The setting for D/A configuration control. This argument is an integer expression formed from one or more of the manifest constants defined in Kdaqdrv.h. There are four group of constants:

#### (1) D/A R/W source selection

KDAQ\_DA\_WRSRC\_Int: Internal timer (default)

KDAQ\_DA\_WRSRC\_AFI0 : From AFI0 pin (**only available for KPXI-AO-4-1M and KPXI-AO-8-1M**)

KDAQ\_DA\_WRSRC\_AFI1 : From AFI1 pin (**NOT available for KPXI-AO-4-1M and KPXI-AO-8-1M**)

KDAQ\_DA\_WRSRC\_SSI: From SSI source

**NOTE** Group (2) DA group selection constant groups are only available for KPXI-AO-4-1M and KPXI-AO-8-1M

### (2) DA group selection

DA\_Group\_A: DA group A  
 DA\_Group\_B : DA group B  
 DA\_Group\_AB: DA group A and group B

### (3) D/A trigger delay counter source selection

KDAQ\_DA\_TDSRC\_Int: Internal timer (default)  
 KDAQ\_DA\_TDSRC\_AFI0 : From AFI0 pin  
 KDAQ\_DA\_TDSRC\_GPTC0: From GPTC0\_OUT pin  
 KDAQ\_DA\_TDSRC\_GPTC1: From GPTC1\_OUT pin

### (4) D/A break delay counter source selection

KDAQ\_DA\_BDSRC\_Int: Internal timer (default)  
 KDAQ\_DA\_BDSRC\_AFI0 : From AFI0 pin  
 KDAQ\_DA\_BDSRC\_GPTC0: From GPTC0\_OUT pin  
 KDAQ\_DA\_BDSRC\_GPTC1: From GPTC1\_OUT pin

When two or more constants are used to form the *ConfigCtrl* argument, the constants are combined with the bitwise-OR operator(`|`).

*TrigCtrl*: The setting for D/A Trigger control. This argument is an integer expression formed from one or more of the manifest constants defined in `kdaqdrvr.h`. There are seven groups of constants:

### (1) Trigger source selection

KDAQ\_DA\_TRGSRC\_SOFT : software (default)  
 KDAQ\_DA\_TRGSRC\_ANA : From analog trigger pin  
 KDAQ\_DA\_TRGSRC\_ExtD: From external digital trigger pin  
 KDAQ\_DA\_TRSRC\_SSI : From SSI source

### (2) Trigger mode selection

KDAQ\_DA\_TRGMOD\_POST : Post Trigger Mode (default)  
 KDAQ\_DA\_TRGMOD\_DELAY : Delay Trigger Mode

**NOTE** Group (3) Re-trigger mode enable constant groups are only available for Post and Delay Trigger Mode.

### (3) Re-trigger mode enable

KDAQ\_DA\_ReTrigEn: Re-trigger in an acquisition is enabled

### (4) Delay2 (break delay) mode enable

KDAQ\_DA\_DLY2En: Delay2/Break delay (the Delay between two consecutive waveform generations) in an acquisition is enabled

**(5) Delay1 Source Selection (only available for Delay Trigger Mode)**

KDAQ\_DA\_Dly1InUI: delay in samples (this value is *not valid* for KPXI-AO-4-1M and KPXI-AO-8-1M)

KDAQ\_DA\_Dly1InTimebase: delay in time base (default)

**(6) Delay2 Source Selection**

KDAQ\_DA\_Dly2InUI: delay in samples (this value is *not valid* for KPXI-AO-4-1M and KPXI-AO-8-1M)

KDAQ\_DA\_Dly2InTimebase: delay in time base (default)

**(7) External Digital Trigger Polarity**

KDAQ\_DA\_TrjPositive: Trigger positive edge active (default)

KDAQ\_DA\_TrjNegative: Trigger negative edge active

When two or more constants are used to form the *TrigCtrl* argument, the constants are combined with the bitwise-OR operator(`|`).

**ReTrgCnt:** The accepted trigger times in an acquisition. If the value of *ReTrgCnt* is 0, the **fixed pattern generation** will be triggered **infinitely**. This argument is only valid for Delay trigger and Post trigger mode. The valid value range is 0 through 65535.

**NOTE** To enable infinite re-trigger mode of fixed pattern generation, call **KDAQ\_AO\_Config** with **KDAQ\_DA\_ReTrigEn** and a zero value for **ReTrgCnt**. To be notified that the pattern generation associated to the next trigger signal of Group AB is complete, you can assign a callback function to **KDAQ\_AO\_EventCallback** with an event type of **DATrigEvent|DATrigEvent\_A|DATrigEvent\_B|DATrigEvent\_AB**.

**DLY1Cnt:** The counter value of DLY1 Counter (the delay time after the trigger signal to the start of the waveform generation) . This argument is only valid for Delay trigger mode. The valid value range is 0 through 65535.

**DLY2Cnt:** The counter value of DLY2 Counter (the **Delay** between two consecutive waveform generations). The valid value range is 0 through 65535.

**NOTE** For KPXI-AO-4-1M and KPXI-AO-8-1M, set *AutoResetBuf* to **FALSE**.

**AutoResetBuf:**

**FALSE:** The DA buffer set by function “KDAQ\_AO\_ContBufferSetup” are retained and must call the function “KDAQ\_AO\_ContBufferReset” to reset the buffer

**TRUE:** The DA buffer set by function “KDAQ\_AO\_ContBufferSetup” are reset automatically by the driver while the AI operation is finishing

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport

## KDAQ\_AO\_ContBufferCompose

**Description** The function fills the data for a specified channel in the buffer for continuous analog output operation. The filled positions of the data in the buffer depend on the type of device. *Except KPXI-AO-4-1M and KPXI-AO-8-1M, this function can only be used for multi-channels of continuous analog output (waveform generation) operation.*

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AO_ContBufferCompose (U16 wCardNumber, U16 group,
    U16 Channel, U32 dwUpdateCount, void *ConBuffer,
    void *Buffer, BOOLEAN fifoload)
```

**Visual Basic**

```
KDAQ_AO_ContBufferCompose (ByVal CardNumber As Integer,
    ByVal group As Integer, ByVal Channel As Integer,
    ByVal WriteCount As Long, ConBuffer As Any, Buffer As Any,
    ByVal fifoload As Byte) As Integer
```

**Parameters** **CardNumber:** The card id of the card to perform this operation.

**group:** The group of analog output channels. The valid values are:

KPXI-SDAQ-4-2M: Not Used  
 KPXI-SDAQ-4-500K: Not Used  
 KPXI-DAQ-64-3M: Not Used  
 KPXI-DAQ-64-500K: Not Used  
 KPXI-DAQ-64-250K: Not Used  
 KPXI-AO-4-1M: DA\_Group\_A  
 KPXI-AO-8-1M: DA\_Group\_A, DA\_Group\_B and DA\_Group\_AB

**Channel:** The AO channel number configured.

KPXI-SDAQ-4-2M : 0 through 1 or All\_Channels (-1)  
 KPXI-SDAQ-4-500K : 0 through 1 or All\_Channels (-1)  
 KPXI-DAQ-64-3M : 0 through 1 or All\_Channels (-1)  
 KPXI-DAQ-64-500K : 0 through 1 or All\_Channels (-1)  
 KPXI-DAQ-64-250K : 0 through 1 or All\_Channels (-1)

KPXI-AO-4-1M : 0 through 3 or All\_Channels (-1)

KPXI-AO-8-1M : 0 through 7 or All\_Channels (-1)

**WriteCount:** The size (in samples) of the buffer of the specified channel (Not the size of the buffer for continuous output operation).

**ConBuffer:** The buffer for continuous output operation.

**Buffer:** the buffer containing the output data for the specified channel.

**fifoload:** The data will be loaded into the on-board DA FIFO by function *KDAQ\_AO\_Group\_FIFOLoad* or not. **This parameter is only valid for KPXI-AO-4-1M and KPXI-AO-8-1M.**

0: won't be loaded into DA fifo

1: will be loaded into DA fifo

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorContIoNotAllowed

## KDAQ\_AO\_ContBufferComposeAll

**Description** The function organizes the data for each channel and stores them in the buffer for continuous analog output operation. The filled positions of the data in the buffer depend on the type of device. *Except KPXI-AO-4-1M and KPXI-AO-8-1M, this function can only be used for multi-channels of continuous analog output (waveform generation) operation.*

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AO_ContBufferComposeAll (U16 wCardNumber,
    U16 group, U32 dwUpdateCount, void *ConBuffer,
    void *Buffer, BOOLEAN fifoload)
```

### Visual Basic

```
KDAQ_AO_ContBufferCompose (ByVal CardNumber As Integer,
    ByVal group As Integer, ByVal WriteCount As Long,
    ConBuffer As Any, Buffer As Any, ByVal fifoload As Byte)
    As Integer
```

**Parameters** **CardNumber:** The card id of the card to perform this operation.

**group:** The group of analog output channels. The valid values are:

KPXI-SDAQ-4-2M: Not Used

KPXI-SDAQ-4-500K: Not Used

KPXI-DAQ-64-3M: Not Used

KPXI-DAQ-64-500K: Not Used

KPXI-DAQ-64-250K: Not Used

KPXI-AO-4-1M: DA\_Group\_A  
 KPXI-AO-8-1M: DA\_Group\_A, DA\_Group\_B and DA\_Group\_AB

**WriteCount:** The size (in samples) of the buffer of the specified channel — not the size of the buffer for continuous output operation.

**ConBuffer:** The buffer for continuous output operation.

**Buffer:** the buffer containing the output data for the specified channel.

**fifoload:** The data will be loaded into on-board DA FIFO by function *KDAQ\_AO\_Group\_FIFOLoad* or not. **This parameter is only valid for KPXI-AO-4-1M and KPXI-AO-8-1M.**

0: won't be loaded into DA fifo

1: will be loaded into DA fifo

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorTransferCountTooLarge , ErrorContloNotAllowed

### KDAQ\_AO\_ContBufferReset

**Description** This function reset all the buffers set by function *KDAQ\_AO\_ContBufferSetup* for continuous analog output. The function has to be called if the data buffers won't be used.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AO_ContBufferReset (U16 wCardNumber)
```

**Visual Basic**

```
KDAQ_AO_ContBufferReset (ByVal CardNumber As Integer)
    As Integer
```

**Parameters** **CardNumber:** The card id of the card to perform this operation.

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorTransferCountTooLarge , ErrorContloNotAllowed

### KDAQ\_AO\_ContBufferSetup

**Description** This function sets up the buffer for continuous analog output operation. The function has to be called repeatedly to setup all of the data buffers (**Except** KPXI-AO-4-1M and KPXI-AO-8-1M, the maximum number of buffers is 2. For KPXI-AO-4-1M and KPXI-AO-8-1M, the maximum number of buffers is 4.).

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AO_ContBufferSetup (U16 wCardNumber,
    void *pwBuffer, U32 dwWriteCount, U16 *BufferId)
```

**Visual Basic**

```
KDAQ_AO_ContBufferSetup (ByVal CardNumber As Integer,
    Buffer As Any, ByVal WriteCount As Long,
    BufferId As Integer) As Integer
```

- Parameters**
- CardNumber:** The card id of the card to perform this operation.
  - Buffer:** The starting address of the memory to contain the output data.
  - WriteCount:** The size (in samples) of the buffer and its value must be even.
  - BufferId:** Returns the index of the buffer currently set up.
- Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorTransferCountTooLarge, ErrorContloNotAllowed

**KDAQ\_AO\_ContStatus**

- Description** While performing continuous D/A conversions, this function is called to get the D/A status.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-AO-4-1M, KPXI-AO-8-1M

- Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AO_ContStatus (U16 CardNumber, U16 *Status)
```

**Visual Basic**

```
KDAQ_AO_ContStatus (ByVal CardNumber As Integer,
    Status As Integer) As Integer
```

- Parameters**
- CardNumber:** The card id of the card to perform this operation.
  - Status:** The continuous AO status returned. The description of the parameter *Status* for various card models is as follows:

Models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K:

bit 0 : '1' indicates D/A FIFO is Underrun  
bit 1 ~ 3 : not used  
bit 4 : '1' indicates D/A FIFO is Empty  
bit 5 : '1' indicates D/A FIFO is Half Full  
bit 6 : '1' indicates D/A FIFO is Full  
bit 7 ~ 15 : not used

Models: KPXI-AO-4-1M / KPXI-AO-8-1M:

bit 0 : '1' indicates D/A FIFO of group A is not Empty  
bit 1 : not used  
bit 2 : '1' indicates D/A FIFO of group A is not Almost Full  
bit 3 : not used  
bit 4 : '1' indicates D/A FIFO of group B is not Empty  
bit 5 : not used  
bit 6 : '1' indicates D/A FIFO of group B is not Almost Full  
bit 7 : not used  
bit 8 ~ 15 : not used



**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered

## KDAQ\_AO\_ContWriteChannel

**Description** This function performs continuous D/A conversions on the specified analog output channel at a rate as close as possible to the rate you specified.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AO_ContWriteChannel (U16 wCardNumber, U16 wChannel,
    U16 BufId, U32 UpdateCount, U32 wIterations, U32 dwCHUI,
    U16 definite, U16 wSyncMode)
```

### Visual Basic

```
KDAQ_AO_ContWriteChannel (ByVal CardNumber As Integer,
    ByVal Channel As Integer, ByVal BufId As Integer,
    ByVal UpdateCount As Long, ByVal Iterations As Long,
    ByVal CHUI As Long, ByVal definite As integer,
    ByVal SyncMode As Integer) As Integer
```

**Parameters** **CardNumber:** The card id of the card to perform this operation.

**Channel:** Analog output channel number

Range: 0 through 1 for KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K

**BufId:** The buffer ID (returned from function *KDAQ\_AO\_ContBufferSetup*) of the buffer containing the acquired data. The size of the buffer with buffer id of *BufId* must have a length (in samples) equal to the value of parameter *UpdateCount*.

**UpdateCount:** If double-buffered mode is disabled, the total update count for each channel to be performed. For double-buffered acquisition, *UpdateCount* is the size (in samples) allocated for each channel in the circular buffer and its value must be a multiple of 2.

**Iterations:** The number of times the data in the buffer is to be output to the port. A value of zero is not allowed. If the DA operation is perform **synchronously**, this argument must be set as 1.

**CHUI:** The length of the Channel Update interval (that is, the counter value between the initiation of each update sequence).

If the timer base is from *external*, the valid range of the value is 8 through 16777215. If the timer base is *Internal timer*, the valid range of the value is as follows:

Range: 40 through 16777215

**definite:** Waveform generation proceeds for a definite time or indefinitely. If double-buffered mode is enabled, this parameter is of no use.

0: indefinitely  
1: definite

**SyncMode:** Whether this operation is performed synchronously or asynchronously. If any trigger mode is enabled by calling `KDAQ_AO_Config()`, this operation should be performed *asynchronously*.

Valid values:

ASYNCH\_OP: asynchronous D/A conversion

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel, ErrorInvalidAdRange, ErrorTransferCountTooLarge, ErrorContIoNotAllowed, ErrorInvalidSampleRate

## KDAQ\_AO\_ContWriteMultiChannels

**Description** This function performs continuous D/A conversions on the specified analog output channels at a rate as close as possible to the rate you specified.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AO_ContWriteMultiChannels (U16 wCardNumber,
    U16 wNumChans, U16 *pwChans, U16 BufId, U32 dwUpdateCount,
    U32 wIterations, U32 dwCHUI, U16 definite, U16 wSyncMode)
```

### Visual Basic

```
KDAQ_AO_ContReadMultiChannels (ByVal CardNumber As Integer,
    ByVal NumChans As Integer, chans As Integer,
    ByVal BufId As Integer, ByVal UpdateCount As Long,
    ByVal Iterations As Long, ByVal CHUI As Long,
    ByVal definite As integer, ByVal SyncMode As Integer)
As Integer
```

**Parameters** **CardNumber:** The card id of the card to perform this operation.

**numChans:** The number of analog input channels in the array *Chans*. The valid value:

KPXI-SDAQ-4-2M: 1 through 2  
 KPXI-SDAQ-4-500K: 1 through 2  
 KPXI-DAQ-64-3M: 1 through 2  
 KPXI-DAQ-64-500K: 1 through 2  
 KPXI-DAQ-64-250K: 1 through 2

**Chans:** Array of analog output channel numbers. The channel order for update data is the same as the order you set in *Chans*.

KPXI-SDAQ-4-2M: numbers in *Chans* must be within 0 and 1  
 KPXI-SDAQ-4-500K: numbers in *Chans* must be within 0 and 1  
 KPXI-DAQ-64-3M: numbers in *Chans* must be within 0 and 1  
 KPXI-DAQ-64-500K: numbers in *Chans* must be within 0 and 1  
 KPXI-DAQ-64-250K: numbers in *Chans* must be within 0 and 1

**BufId:** The buffer ID (returned from function *KDAQ\_AO\_ContBufferSetup*) of the buffer containing the output data. The size of the buffer with buffer id of *BufId* must have a length equal to or greater than the value of *WriteCount X numChans*.

The data order in the buffer is in an interleaved sequence as follows. So the data for channel 0 is stored in *Buffer[0], Buffer[2], Buffer[4], ...* The data for channel 1 is stored in *Buffer[1], Buffer[3], Buffer[5], ...*

**UpdateCount:** If double-buffered mode is disabled, the total update count for each channel to be performed. For double-buffered acquisition, *UpdateCount* is the size (in samples) allocated for each channel in the circular buffer and its value must be a multiple of 2.

**Iterations:** The number of times the data in the buffer is to be output to the port. A value of zero is not allowed. If the DA operation is perform **synchronously**, this argument must be set as 1.

**CHUI:** The length of the Channel Update interval (that is, the counter value between the initiation of each update sequence).

If the timer base is from *external*, the valid range of the value is 8 through 16777215. If the timer base is *Internal timer*, the valid range of the value is as follows:

Range: 40 through 16777215

**definite:** Waveform generation proceeds for a definite time or indefinitely. If double-buffered mode is enabled, this parameter is of no use.

0: indefinitely  
1: definite

**SyncMode:** Whether this operation is performed synchronously or asynchronously. If any trigger mode is enabled by calling *KDAQ\_AO\_Config()*, this operation should be performed **asynchronously**.

Valid values:

ASYNCH\_OP: asynchronous D/A conversion

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel, ErrorInvalidSampleRate, ErrorInvalidAdRange, ErrorTransferCountTooLarge, ErrorContIoNotAllowed

### KDAQ\_AO\_DelayTrig\_Config

**Description** Informs KDAQ-DRVR library of the update clock source and the trigger properties for the Keithley PXI DAQ device that performs delay triggered waveform generation operation.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax****Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AO_DelayTrig_Config (U16 wCardNumber, U16 ClkSrc,
    U16 TrigSrcCtrl, U16 DLY1Cnt, U16 DLY2Ctrl, U16 DLY2Cnt,
    U16 ReTrgEn, U16 ReTrgCnt, BOOLEAN AutoResetBuf)
```

**Visual Basic**

```
KDAQ_AO_DelayTrig_Config (ByVal CardNumber As Integer,
    ByVal ClkSrc As Integer, ByVal TrigSrcCtrl As Integer,
    ByVal DLY1Cnt As Integer, ByVal DLY2Ctrl As Integer,
    ByVal DLY2Cnt As Integer, ByVal ReTrgEn As Integer,
    ByVal ReTrgCnt As Integer, ByVal AutoResetBuf As Byte)
As Integer
```

**Parameters**

**CardNumber:** The card id of the card to perform this operation.

**ClkSrc:** The setting for D/A update clock source. This argument is an integer expression formed from one or more of the manifest constants defined in Kdaqdrv.h. There are four groups of constants:

**(1) D/A R/W source selection**

KDAQ\_DA\_WRSRC\_Int: Internal timer (default)  
 KDAQ\_DA\_WRSRC\_AFI0 : From AFIO pin  
 KDAQ\_DA\_WRSRC\_SSI: From SSI source

The following constant groups are only available in Model KPXI-AO-4-1M and Model KPXI-AO-8-1M

**(2) DA group selection**

DA\_Group\_A: DA group A  
 DA\_Group\_B : DA group B  
 DA\_Group\_AB: DA group A and group B

**(3) D/A trigger delay counter source selection**

KDAQ\_DA\_TDSRC\_Int: Internal timer (default)  
 KDAQ\_DA\_TDSRC\_AFI0 : From AFIO pin  
 KDAQ\_DA\_TDSRC\_GPTC0: From GPTC0\_OUT pin  
 KDAQ\_DA\_TDSRC\_GPTC1: From GPTC1\_OUT pin

**(4) D/A break delay counter source selection**

KDAQ\_DA\_BDSRC\_Int: Internal timer (default)  
 KDAQ\_DA\_BDSRC\_AFI0 : From AFIO pin

KDAQ\_DA\_BDSRC\_GPTC0: From GPTC0\_OUT pin  
 KDAQ\_DA\_BDSRC\_GPTC1: From GPTC1\_OUT pin

When two or more constants are used to form the *ConfigCtrl* argument, the constants are combined with the bitwise-OR operator(`|`).

**TrigSrcCtrl:** The setting for D/A Trigger control. This argument is an integer expression formed from one or more of the manifest constants defined in `kdaqdrvr.h`. There are three groups of constants:

**(1) Trigger source selection**

KDAQ\_DA\_TRGSRC\_SOFT : software (default)  
 KDAQ\_DA\_TRGSRC\_ANA : From analog trigger pin  
 KDAQ\_DA\_TRGSRC\_ExtD: From external digital trigger pin  
 KDAQ\_DA\_TRSRC\_SSI : From SSI source

**(2) Delay1 source selection**

KDAQ\_DA\_Dly1InUI: delay in samples (this value is *not valid* for KPXI-AO-4-1M and KPXI-AO-8-1M)

KDAQ\_DA\_Dly1InTimebase: delay in time base (default)

**(3) External digital trigger polarity**

KDAQ\_DA\_TrgPositive: Trigger positive edge active (default)  
 KDAQ\_DA\_TrgNegative: Trigger negative edge active  
 When two or more constants are used to form the *TrigSrcCtrl* argument, the constants are combined with the bitwise-OR operator(`|`).

**DLY1Cnt:** The counter value of DLY1 Counter (the delay time after the trigger signal to the start of the waveform generation). The valid value range is 0 through 65535.

**DLY2Ctrl:** The setting for D/A Trigger control. This argument is an integer expression formed from one or more of the manifest constants defined in `kdaqdrvr.h`. There are two groups of constants:

**(1) delay2 (break delay) mode enable**

KDAQ\_DA\_DLY2En: Delay2/Break delay (the Delay between two consecutive waveform generations) in an acquisition is enabled

**(2) Delay2 Source Selection**

KDAQ\_DA\_Dly2InUI: delay in samples (this value is *not valid* for KPXI-AO-4-1M and KPXI-AO-8-1M)

KDAQ\_DA\_Dly2InTimebase: delay in time base (default)

When two or more constants are used to form the *DLY2Ctrl* argument, the constants are combined with the bitwise-OR operator(`|`).

**DLY2Cnt:** The counter value of DLY2 Counter (the **Delay** between two consecutive waveform generations). The valid value range is 0 through 65535.

**ReTrgEn:**

0: Re-trigger in an acquisition is disabled. (default value)

1: Re-trigger in an acquisition is enabled.

**ReTrgCnt:** The accepted trigger times in an acquisition. The valid value range is 0 through 65535.

**AutoResetBuf:**

FALSE: The DA buffer set by function "KDAQ\_AO\_ContBufferSetup" are retained and must call the function "KDAQ\_AO\_ContBufferReset" to reset the buffer

TRUE: The DA buffer set by function "KDAQ\_AO\_ContBufferSetup" are reset automatically by the driver while the AI operation is finishing

**NOTE** For KPXI-AO-4-1M and KPXI-AO-8-1M, set **AutoResetBuf** to **FALSE**.

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport

## KDAQ\_AO\_EventCallback

**Description** Controls and notifies the user's application when a specified DAQ event occurs. The notification is performed through a user-specified callback function.

For windows version, the event message will be removed automatically after calling *KDAQ\_AO\_Async\_Clear* or *KDAQ\_AO\_Group\_WFM\_AsyncClear*. The event message can also be manually removed by set the parameter "mode" to be 0.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AO_EventCallback (U16 CardNumber, I16 mode,
    I16 EventType, U32 callbackAddr)
```

**Visual Basic 5**

```
KDAQ_AO_EventCallback (ByVal CardNumber As Integer,
    ByVal mode As Integer, ByVal EventType As Integer,
    ByVal callbackAddr As Long) As Integer
```

**Parameters** **CardNumber:** The card id of the card to perform this operation.

**mode:** add or remove the event message. The valid values:

0: remove

1: add

**EventType:** event criteria. The valid values are as follows:

**For KPXI-SDAQ-4-500K, KPXI-SDAQ-4-2M, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K:**

DBEvent: Notification for the next half buffer of data in circular buffer is ready for transfer

DAQEnd: Notification for the completeness of asynchronous analog output operation

DATrigEvent: Notification for the pattern generation associated with the next trigger signal is completed

**For KPXI-AO-4-1M and KPXI-AO-8-1M:**

DBEvent : Notification for the next half buffer of data in circular buffer is ready for transfer

DAQEnd\_A: Notification for the completeness of asynchronous analog output operation of Group A

DAQEnd\_B: Notification for the completeness of asynchronous analog output operation of Group B

DAQEnd\_AB: Notification for the completeness of asynchronous analog output operation of Group AB

DATrigEvent\_A: Notification for the pattern generation associated to the next trigger signal of Group A is completed

DATrigEvent\_B: Notification for the pattern generation associated to the next trigger signal of Group B is completed

DATrigEvent\_AB: Notification for the pattern generation associated to the next trigger signal of Group AB is completed

**callbackAddr:** the address of the user callback function. KDAQ-DRVR calls this function when the specified event occurs. If you wish to remove the event message, set *callbackAddr* to 0.

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport

**KDAQ\_AO\_Group\_FIFOLoad**

**Description** Loads a waveform buffer to on-board DA FIFOs

This function is supported by the following models: KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

I16 KDAQ\_AO\_Group\_FIFOLoad (U16 wCardNumber, U16 group, U16 BufId, U32 UpdateCount)

**Visual Basic**

KDAQ\_AO\_Group\_FIFOLoad (ByVal CardNumber As Integer, ByVal group As Integer, ByVal BufId As Integer, ByVal UpdateCount As Long) As Integer

**Parameters** **CardNumber:** The card id of the card to perform this operation.

**group:** The group of analog output channels. The valid value:

KPXI-AO-4-1M: DA\_Group\_A

KPXI-AO-8-1M: DA\_Group\_A, DA\_Group\_B and DA\_Group\_AB

**BufId:** The buffer ID (returned from function *KDAQ\_AO\_ContBufferSetup*) of the buffer containing the output data. **The size of the buffer with id of *BufId* must have a length equal to or smaller than the size of FIFOs on board.**

KPXI-AO-4-1M: DA\_Group\_A: 8K samples

KPXI-AO-8-1M: DA\_Group\_A: 8K samples

DA\_Group\_B: 8K samples

DA\_Group\_AB: 16K samples

DA\_Group\_A: 8K samples

The sequence of the data in the buffer is the same as the sequence of the channels in the specified group.

For example:

DA\_Group\_A : channel 0, 1 enabled

DA\_Group\_B : channel 4, 5 enabled, and group loaded is DA\_Group\_AB, Then

The data for channel 0 is in *Buffer*[0], *Buffer*[4], *Buffer*[8], ...

The data for channel 1 is in *Buffer*[1], *Buffer*[5], *Buffer*[9], ...

The data for channel 4 is in *Buffer*[2], *Buffer*[6], *Buffer*[10], ...

The data for channel 5 is in *Buffer*[3], *Buffer*[7], *Buffer*[11], ...

The valid data range is within 0 to 4095.

**UpdateCount:** The count of data loaded to the FIFOs.

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel,

## KDAQ\_AO\_Group\_Setup

**Description** Assigns one or more analog output channels to a waveform generation group.

This function is supported by the following models: KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AO_Group_Setup (U16 wCardNumber, U16 group,
    U16 wNumChans, U16 *Chans)
```



**Visual Basic**

```
KDAQ_AO_Group_Setup (ByVal CardNumber As Integer,
    ByVal group As Integer, ByVal wNumChans As Integer,
    Chans As Integer) As Integer
```

**Parameters**

**CardNumber:** The card id of the card to perform this operation.

**numChans:** The number of analog output channels in the array *Chans*. The valid value:

KPXI-AO-4-1M: 1 through 4  
 KPXI-AO-8-1M: 1 through 8

**group:** The group of analog output channels. The valid values are as follows:

KPXI-AO-4-1M: DA\_Group\_A  
 KPXI-AO-8-1M: DA\_Group\_A, DA\_Group\_B and DA\_Group\_AB

**Chans:** Array of analog output channel numbers. The channel order for update data is the same as the order you set in *Chans*.

KPXI-AO-4-1M: numbers in *Chans* must be:

DA\_Group\_A : 0~ 3

KPXI-AO-8-1M: numbers in *Chans* must be:

DA\_Group\_A : 0~ 3  
 DA\_Group\_B : 4 ~ 7  
 DA\_Group\_AB : 0 ~ 7

**Return Value**

NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel,

**KDAQ\_AO\_Group\_Update**

**Description**

Writes binary values to the specified group of analog output channels simultaneously.

This function is supported by the following models: KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax**

**Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AO_Group_Update (U16 CardNumber, U16 group,
    I16 *Buffer)
```

**Visual Basic**

```
KDAQ_AO_Group_Update (ByVal CardNumber As Integer,
    ByVal group As Integer, Buffer As Integer) As Integer
```

- Parameters**    **CardNumber:** The card id of the card to perform this operation.  
**group:** The group of analog output channels. The valid value:  
 KPXI-AO-4-1M: DA\_Group\_A  
 KPXI-AO-8-1M: DA\_Group\_A, DA\_Group\_B and DA\_Group\_AB
- Buffer:** An integer array to contain the update data. The length (in samples) of *Buffer* must be equal to or greater the total number of channels in the specified DA group. The range of value to be written to the analog output channels is 0 through 4095.
- Return Value**    NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel

### KDAQ\_AO\_Group\_VUpdate

**Description**    Accepts voltage values, scales them to the proper binary values and writes binary values to the specified group of analog output channels simultaneously.

This function is supported by the following models: KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax**            **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AO_Group_VUpdate (U16 CardNumber, U16 group,
    F64 *Voltage)
```

#### Visual Basic

```
KDAQ_AO_Group_VUpdate (ByVal CardNumber As Integer,
    ByVal group As Integer, Voltage As Double) As Integer
```

- Parameters**    **CardNumber:** The card id of the card to perform this operation.  
**group:** The group of analog output channels. The valid values are as follows:  
 KPXI-AO-4-1M: DA\_Group\_A  
 KPXI-AO-8-1M: DA\_Group\_A, DA\_Group\_B and DA\_Group\_AB

**Voltage:** An floating-point voltage value array to contain the update data. The length (in samples) of *Voltage* must be equal to or greater the total number of channels in the specified DA group. The range of voltages depends on the type of device, on the output polarity, and on the voltage reference (external or internal).

**Return Value**    NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel

### KDAQ\_AO\_Group\_WFM\_AsyncCheck

**Description**    Check the current status of the asynchronous analog output operation of a specified group. This function is only available for the device that uses timer pacer (**KDAQ\_DA\_WRSRC\_Int**) as the D/A R/W Source.

This function is supported by the following models: KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax**            **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AO_Group_WFM_AsyncCheck (U16 CardNumber, U16 group,
    U8 *Stopped, U32 *WriteCnt)
```

**Visual Basic**

KDAQ\_AO\_Group\_WFM\_AsyncCheck (ByVal CardNumber As Integer, ByVal group As Integer, Stopped As Byte, WriteCnt As Long) As Integer

**Parameters** **CardNumber:** The card id of the card that performs the asynchronous operation.

**group:** The group of analog output channels. The valid values are as follows:

KPXI-AO-4-1M: DA\_Group\_A  
 KPXI-AO-8-1M: DA\_Group\_A, DA\_Group\_B and DA\_Group\_AB

**Stopped:** Whether the asynchronous analog output operation has completed. DA\_Group\_A and DA\_Group\_B:

If *Stopped* = 1, the analog output operation has stopped. Either the number of D/A conversions indicated in the call that initiated the asynchronous analog output operation has completed or an error has occurred. If *Stopped* = 0, the operation is not yet complete.

DA\_Group\_AB:

Bit0: asynchronous analog output operation of group A  
 Bit1: asynchronous analog output operation of group B

If *Stopped* = 3, the analog output operation has stopped for both DA group A and group B.

If *Stopped* = 1, the analog output operation has stopped for DA group A.

If *Stopped* = 2, the analog output operation has stopped for DA group B.

If *Stopped* = 0, the operation is not yet complete for both DA group A and group B.

**WriteCnt:** The number of analog output data that have been written at the time of calling KDAQ\_AO\_Group\_WFM\_AsyncCheck ().

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport

**KDAQ\_AO\_Group\_WFM\_AsyncClear**

**Description** Software terminates the asynchronous analog output operation of a specified group.

This function is supported by the following models: KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

l16 KDAQ\_AO\_Group\_WFM\_AsyncClear (U16 CardNumber, U16 group, U32 \*WriteCnt, U16 stop\_mode)

**Visual Basic**

KDAQ\_AO\_Group\_WFM\_AsyncClear (ByVal CardNumber As Integer, ByVal group As Integer, WriteCnt As Long, ByVal stop\_mode As Integer) As Integer

- Parameters**
- CardNumber:** The card id of the card that performs the asynchronous operation.
  - group:** The group of analog output channels. The valid values are as follows:  
  
 KPXI-AO-4-1M: DA\_Group\_A  
 KPXI-AO-8-1M: DA\_Group\_A, DA\_Group\_B and DA\_Group\_AB
  - WriteCnt:** The number of analog output data that have been written at the time of calling KDAQ\_AO\_Group\_WFM\_AsyncClear ().
  - stop\_mode:** The DA transfer termination mode selected. Valid values are as follows:  
  
 KDAQ\_DA\_TerminateImmediate: Software terminate the DA continuous operation immediately  
  
 KDAQ\_DA\_TerminateUC: Software terminate the DA continuous operation on next update counter terminal count  
  
 KDAQ\_DA\_TerminateIC: Software terminate the DA continuous operation on iteration count
- Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport

## KDAQ\_AO\_Group\_WFM\_Start

- Description** This function performs continuous D/A conversions on the specified group of analog output channels at a rate as close as possible to the rate you specified. This function is supported by the following models: KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AO_Group_WFM_Start (U16 wardNumber, U16 group,
    U16 fstBufIdOrNotUsed, U16 sndBufId, U32 dwUpdateCount,
    U32 wIterations, U32 dwCHUI, U16 definite)
```

**Visual Basic**

```
KDAQ_AO_Group_WFM_Start (ByVal CardNumber As Integer,
    ByVal group As Integer,
    ByVal fstBufIdOrNotUsed As Integer,
    ByVal sndBufId As Integer, ByVal UpdateCount As Long,
    ByVal Iterations As Long, ByVal CHUI As Long,
    ByVal definite As Integer) As Integer
```

**Parameters**

**CardNumber:** The card id of the card to perform this operation.

**group:** The group of analog output channels. The valid values are as follows:

KPXI-AO-4-1M: DA\_Group\_A

KPXI-AO-8-1M: DA\_Group\_A, DA\_Group\_B and DA\_Group\_AB

**FstBufIdOrNotUsed:** If the data have been loaded by the function *KDAQ\_AO\_Group\_FIFOLoad*, the value of *fstBufIdOrNotUsed* must be **BufferNotUsed**.

If the value of the parameter *fstBufIdOrNotUsed* is not *BufferNotUsed*, or the data have not been loaded, the value of *fstBufIdOrNotUsed* is the buffer ID (returned from function *KDAQ\_AO\_ContBufferSetup*) of the first buffer containing the output data.

For KPXI-AO-4-1M only, the data are transferred to the DA FIFOs from the buffer with id of *fstBufIdOrNotUsed* through DMA operation. **The sequence of the data in the buffer is the same as the sequence of the channels in the group.**

For example:

DA\_Group\_A : channel 0, 1, 2 enabled

The data for channel 0 is in *Buffer[0]*, *Buffer[4]*, *Buffer[8]*, ...

The data for channel 1 is in *Buffer[1]*, *Buffer[5]*, *Buffer[9]*, ...

The data for channel 2 is in *Buffer[2]*, *Buffer[6]*, *Buffer[10]*, ...

For KPXI-AO-8-1M only, the data are transferred to the DA FIFOs from the buffer with id of *fstBufIdOrNotUsed* through 32-bit DMA operation. The upper 16-bits of the data is for the FIFO of Group B and the lower 16 bit of the data is for the FIFO of Group A.

If the group is **DA\_Group\_A** or **DA\_Group\_B**, the 16-bit buffer must contain **two times of the total update samples** because of 32-bit DMA data transfer. The data points for the specified group are in the even elements of the buffer.

**(1) DA\_Group\_A : channel 0, 1 enabled**

The data for channel 0 is in *Buffer[0]*, *Buffer[4]*, *Buffer[8]*, ...

The data for channel 1 is in *Buffer[2]*, *Buffer[6]*, *Buffer[10]*, ...

**(2). DA\_Group\_B : channel 4, 5 enabled**

The data for channel 4 is in *Buffer[1]*, *Buffer[5]*, *Buffer[9]*, ...

The data for channel 5 is in *Buffer[3]*, *Buffer[7]*, *Buffer[11]*, ...

If the group is **DA\_Group\_AB**, the buffer must contain **the same sample counts** for group A and group B. The data for each group is in interleaved sequence.

For example:

DA\_Group\_A : channel 0, 1 enabled

DA\_Group\_B : channel 4, 5 enabled, and group loaded is DA\_Group\_AB,

Then the data for channel 0 is in *Buffer[0]*, *Buffer[4]*, *Buffer[8]*, ...

the data for channel 4 is in *Buffer[1]*, *Buffer[5]*, *Buffer[9]*, ...

the data for channel 1 is in *Buffer[2]*, *Buffer[6]*, *Buffer[10]*, ...  
 the data for channel 5 is in *Buffer[3]*, *Buffer[7]*, *Buffer[11]*, ...

**sndBufId:** The buffer ID (returned from function *KDAQ\_AO\_ContBufferSetup*) of the second buffer containing the output data. **This parameter is only available for double buffer mode of waveform generation.**

**UpdateCount:** If the data have been loaded by the function *KDAQ\_AO\_Group\_FIFOLoad*, the parameter of UpdateCount is **of no use**. If the data is transferred though DMA operation and double-buffered mode is disabled, the value of *UpdateCount* is the total update count for each channel to be performed. For double-buffered acquisition, *UpdateCount* is the size (in samples) allocated for each channel in the circular buffer and its value must be a multiple of 2.

**Iterations:** The number of times the data in the buffer is to be output to the port. A value of zero is not allowed.

**CHUI:** The length of the Channel Update interval (that is, the counter value between the initiation of each update sequence).

If the timer base is from *external*, the valid range of the value is 8 through 16777215. If the timer base is *Internal timer*, the valid range of the value is as follows:

Range: 40 through 16777215

**definite:** Waveform generation proceeds for a definite time or indefinitely. If double-buffered mode is enabled, this parameter is of no use.

0: indefinitely

1: definite

**NOTE** *If FIFO mode of waveform generation is enabled, the double-buffered waveform generation is not allowed.*

**NOTE** *If the group DA\_Group\_AB is specified, the iterations and scan rate are the same for both groups.*

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel, ErrorInvalidSampleRate, ErrorTransferCountTooLarge, ErrorContIoNotAllowed

## KDAQ\_AO\_Group\_WFM\_StopConfig

**Description** Informs KDAQ-DRVR library of the stop source and the stop mode for the asynchronous analog output operation of a specified group.

This function is supported by the following models: KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

116 KDAQ\_AO\_Group\_WFM\_StopConfig (U16 wCardNumber, U16 group, U16 stopSrc, U16 stopMode)

**Visual Basic**

KDAQ\_AO\_Group\_WFM\_StopConfig (ByVal CardNumber As Integer, ByVal group As Integer, ByVal stopSrc As Integer, ByVal stopMode As Integer) As Integer

**Parameters** **CardNumber:** The card id of the card that performs the asynchronous operation.

**group:** The group of analog output channels. Valid values:  
 KPXI-AO-4-1M: DA\_Group\_A  
 KPXI-AO-8-1M: DA\_Group\_A, DA\_Group\_B and DA\_Group\_AB

**stopSrc:** The DA transfer termination source selected.  
 KDAQ\_DA\_STOPSRC\_SOFT: Software terminate the DA continuous operation  
 KDAQ\_DA\_STOPSRC\_AFI0: Terminate the DA continuous operation from the external signal of AFI0  
 KDAQ\_DA\_STOPSRC\_ATrig: Terminate the DA continuous operation from the external signal of analog trigger  
 KDAQ\_DA\_STOPSRC\_AFI1: Terminate the DA continuous operation from the external signal of AFI1

**stop\_mode:** The DA transfer termination mode selected. Valid values:  
 KDAQ\_DA\_TerminateImmediate : Terminate the DA continuous operation immediately  
 KDAQ\_DA\_TerminateUC: Terminate the DA continuous operation on next update counter terminal count  
 KDAQ\_DA\_TerminateIC: Terminate the DA continuous operation on iteration count

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel, ErrorInvalidSampleRate, ErrorTransferCountTooLarge, ErrorContIoNotAllowed

**KDAQ\_AO\_InitialMemoryAllocated**

**Description** This function returns the available memory size for analog output in the device driver in argument *MemSize*. The continuous analog output transfer size can not exceed this size.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AO_InitialMemoryAllocated (U16 CardNumber,
    U32 *MemSize)
```

**Visual Basic**

KDAQ\_AO\_InitialMemoryAllocated (ByVal CardNumber As Integer, MemSize As Long) As Integer

**Parameters** **CardNumber:** The card id of the card to perform this operation.

**MemSize:** The available memory size for continuous AO in device driver of this card. The unit is KB (1024 bytes).

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered

## KDAQ\_AO\_PostTrig\_Config

**Description** Informs KDAQ-DRVR library of the update clock source and the trigger properties for the Keithley PXI DAQ device that performs post triggered waveform generation operation.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AO_PostTrig_Config (U16 wCardNumber, U16 ClkSrc,
    U16 TrigSrcCtrl, U16 DLY2Ctrl, U16 DLY2Cnt, U16 ReTrgEn,
    U16 ReTrgCnt, BOOLEAN AutoResetBuf)
```

### Visual Basic

```
KDAQ_AO_PostTrig_Config (ByVal CardNumber As Integer,
    ByVal ClkSrc As Integer, ByVal TrigSrcCtrl As Integer,
    ByVal DLY2Ctrl As Integer, ByVal DLY2Cnt As Integer,
    ByVal ReTrgEn As Integer, ByVal ReTrgCnt As Integer,
    ByVal AutoResetBuf As Byte) As Integer
```

**Parameters** **CardNumber:** The card id of the card to perform this operation.

**ClkSrc:** The setting for D/A update clock source. This argument is an integer expression formed from one or more of the manifest constants defined in **kdaqdrv.h**. There are three groups of constants:

(1) D/A R/W Source Selection

KDAQ\_DA\_WRSRC\_Int: Internal timer (default)  
 KDAQ\_DA\_WRSRC\_AFI0 : From AFI0 pin  
 KDAQ\_DA\_WRSRC\_SSI: From SSI source

(2) DA group Selection (only available for KPXI-AO-4-1M and KPXI-AO-8-1M)

DA\_Group\_A: DA group A  
 DA\_Group\_B : DA group B  
 DA\_Group\_AB: DA group A and group B

(3) D/A Break delay Counter Source Selection

KDAQ\_DA\_BDSRC\_Int: Internal timer (default)  
 KDAQ\_DA\_BDSRC\_AFI0 : From AFI0 pin  
 KDAQ\_DA\_BDSRC\_GPTC0: From GPTC0\_OUT pin  
 KDAQ\_DA\_BDSRC\_GPTC1: From GPTC1\_OUT pin

When two or more constants are used to form the *ConfigCtrl* argument, the constants are combined with the bitwise-OR operator(**|**).

**TrigSrcCtrl:** The setting for D/A Trigger control. This argument is an integer expression formed from one or more of the manifest constants defined in **kdaqdrv.h**. There are two groups of constants:

(1) Trigger Source Selection

KDAQ\_DA\_TRGSRC\_SOFT : software (default)  
 KDAQ\_DA\_TRGSRC\_ANA : From analog trigger pin  
 KDAQ\_DA\_TRGSRC\_ExtD: From external digital trigger pin  
 KDAQ\_DA\_TRSRC\_SSI : From SSI source



- (2) External Digital Trigger Polarity
  - KDAQ\_DA\_TrgPositive: Trigger positive edge active (default)
  - KDAQ\_DA\_TrgNegative: Trigger negative edge active

When two or more constants are used to form the *TrigSrcCtrl* argument, the constants are combined with the bitwise-OR operator(`|`).

***DLY2Ctrl***: The setting for D/A Trigger control. This argument is an integer expression formed from one or more of the manifest constants defined in `kdaqdrvr.h`. There are two groups of constants:

- (1) Delay2 (Break delay) Mode Enable
  - KDAQ\_DA\_DLY2En: Delay2/Break delay (the Delay between two consecutive waveform generations) in an acquisition is enabled
- (2) Delay2 Source Selection
  - KDAQ\_DA\_Dly2InUI: delay in samples (this value is **not valid** for KPXI-AO-4-1M, KPXI-AO-8-1M)
  - KDAQ\_DA\_Dly2InTimebase: delay in time base (default)

When two or more constants are used to form the *DLY2Ctrl* argument, the constants are combined with the bitwise-OR operator(`|`).

***DLY2Cnt***: The counter value of DLY2 Counter (the **Delay** between two consecutive waveform generations). The valid value range is 0 through 65535.

***ReTrgEn***: 0: Re-trigger in an acquisition is disabled. (default value)  
 1: Re-trigger in an acquisition is enabled.

***ReTrgCnt***: The accepted trigger times in an acquisition. The valid value range is 0 through 65535.

***AutoResetBuf***: For KPXI-AO-4-1M and KPXI-AO-8-1M, set this parameter to FALSE.

FALSE: The DA buffer set by function "KDAQ\_AO\_ContBufferSetup" are retained and must call the function "KDAQ\_AO\_ContBufferReset" to reset the buffer

TRUE: The DA buffer set by function "KDAQ\_AO\_ContBufferSetup" are reset automatically by the driver while the AI operation is finishing

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport

### KDAQ\_AO\_SimuWriteChannel

**Description** Writes binary values to the specified analog output channels simultaneously.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AO_SimuWriteChannel (U16 wCardNumber,
                             U16 wNumChans, U16 *pwBuffer)
```

**Visual Basic**

```
KDAQ_AO_SimuWriteChannel (ByVal CardNumber As Integer,
    ByVal NumChans As Integer, Buffer As Integer) As Integer
```

**Parameters** **CardNumber:** The card id of the card to perform this operation.

**numChans:** The number of analog output channels. The valid value:

KPXI-SDAQ-4-2M: 1 through 2  
 KPXI-SDAQ-4-500K: 1 through 2  
 KPXI-DAQ-64-3M: 1 through 2  
 KPXI-DAQ-64-500K: 1 through 2  
 KPXI-DAQ-64-250K: 1 through 2

**Buffer:** An integer array to contain the update data. The length (in samples) of *Buffer* must be equal to or greater the value of parameter *numChans*. The range of value to be written to the analog output channels:

Range: 0 – 4095 for KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, and KPXI-DAQ-64-250K

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel

**KDAQ\_AO\_VoltScale**

**Description** Scales a voltage (or a current value) to a binary value.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_AO_VoltScale (U16 CardNumber, U16 Channel,
    F64 Voltage, I16 *binValue)
```

**Visual Basic**

```
KDAQ_AO_VoltScale (ByVal CardNumber As Integer,
    ByVal Channel As Integer, ByVal Voltage As Double,
    binValue As Integer) As Integer
```

**Parameters** **CardNumber:** The card id of the card to perform this operation.

**Channel:** The analog output channel number.

Range: 0 or 1 for KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-64-3M

Range: 0 – 3 for KPXI-AO-4-1M

Range: 0 – 7 for KPXI-AO-8-1M

**Voltage:** Voltage, in volts, to be converted to a binary value

**binValue:** the converted binary value returned

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel, ErrorDaVoltageOutOfRange

**KDAQ\_AO\_VWriteChannel**

**Description** Accepts a voltage value (or a current value), scales it to the proper binary value and writes a binary value to the specified analog output channel.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K

**Syntax**      **Microsoft C/C++ and Borland C++**

I16 KDAQ\_AO\_VWriteChannel (U16 CardNumber, U16 Channel, F64 Voltage)

**Visual Basic**

KDAQ\_AO\_VWriteChannel (ByVal CardNumber As Integer, ByVal Channel As Integer, ByVal Voltage As Double) As Integer

**Parameters**      **CardNumber:** The card id of the card to perform this operation.

**Channel:** The analog output channel number:

- Range:              0 or 1 for KPXI-SDAQ-4-2M
- Range:              0 or 1 for KPXI-SDAQ-4-500K
- Range:              0 or 1 for KPXI-DAQ-64-3M
- Range:              0 or 1 for KPXI-DAQ-64-500K
- Range:              0 or 1 for KPXI-DAQ-64-250K

**Voltage:** The value to be scaled and written to the analog output channel. The range of voltages depends on the type of device, on the output polarity, and on the voltage reference (external or internal).

**Return Value**      NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel, ErrorDaVoltageOutOfRange

**KDAQ\_AO\_WriteChannel**

**Description**      Writes a binary value to the specified analog output channel.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K

**Syntax**      **Microsoft C/C++ and Borland C++**

I16 KDAQ\_AO\_WriteChannel (U16 CardNumber, U16 Channel, I16 Value)

**Visual Basic**

KDAQ\_AO\_WriteChannel (ByVal CardNumber As Integer, ByVal Channel As Integer, ByVal Value As Integer) As Integer

**Parameters**      **CardNumber:** The card id of the card to perform this operation.

**Channel:** The analog output channel number.

Range: 0 or 1 for KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-64-3M

**Value:** The value to be written to the analog output channel.

Range: 0 – 4095 for KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-64-3M

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel

### KDAQ\_DB\_Auto\_Calibration\_ALL

**Description** Uses this function to calibrate your Keithley PXI DAQ device. When the function is called, the device goes into a self-calibration cycle. The function does not return until the self-calibration is completed.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_DB_Auto_Calibration_ALL(U16 CardNumber)
```

#### Visual Basic

```
KDAQ_DB_Auto_Calibration_ALL (ByVal CardNumber As Integer)
    As Integer
```

**Parameters** **CardNumber:** The card id of the card to perform this operation.

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel

### KDAQ\_DI\_ReadLine

**Description** Read the digital logic state of the specified digital line in the specified port.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_DI_ReadLine (U16 CardNumber, U16 Port, U16 Line,
    U16 *State)
```

#### Visual Basic

```
KDAQ_DI_ReadLine (ByVal CardNumber As Integer,
    ByVal Port As Integer, ByVal Line As Integer,
    Value As Integer) As Integer
```

**Parameters** **CardNumber:** The card id of the card to perform this operation.

**Port:** Digital input port number. Valid values:  
Channel\_P1A, Channel\_P1B,  
Channel\_P1C, Channel\_P1CL,  
Channel\_P1CH

**Line:** The digital line to be read. The valid value is 0 through 7

**State:** Returns the digital logic state, 0 or 1, of the specified line.

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel

### KDAQ\_DI\_ReadPort

**Description** Read digital data from the specified digital input port.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_DI_ReadPort (I16 CardNumber, U16 Port, U32 *Value)
```

**Visual Basic**

```
KDAQ_DI_ReadPort (ByVal CardNumber As Integer,
    ByVal Port As Integer, Value As Long) As Integer
```

**Parameters** **CardNumber:** The card id of the card to perform this operation.  
**Port:** Digital input port number. The valid value:

Channel\_P1A,Channel\_P1B,  
 Channel\_P1C,Channel\_P1CL,  
 Channel\_P1CH

**Value:** Returns the digital data read from the specified port. The returned value is 8-bit data.

**Return Value** NoError, CardNotRegistered, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport

**KDAQ\_DIO\_PortConfig**

**Description** Informs KDAQ-DRVR library of the port selected and the direction (Input or output) setting of the selected port.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_DIO_PortConfig (U16 CardNumber, U16 Port, U16
    Direction)
```

**Visual Basic**

```
KDAQ_DIO_PortConfig (ByVal CardNumber As Integer,
    ByVal Port As Integer, ByVal Direction As Integer)
    As Integer
```

**Parameters** **CardNumber:** The card id of the card to perform this operation.

**Port:** The port selected. Valid values:

Channel\_P1A,Channel\_P1B,  
 Channel\_P1C,Channel\_P1CL  
 Channel\_P1CH

**Direction:** The port direction of DIO port. Valid values:

INPUT\_PORT  
 OUTPUT\_PORT

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel

## KDAQ\_DO\_ReadLine

**Description** Read back the digital logic state of the specified digital output line in the specified port.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

I16 KDAQ\_DO\_ReadLine (U16 CardNumber, U16 Port, U16 Line, U16 \*State)

### Visual Basic

KDAQ\_DO\_ReadLine (ByVal CardNumber As Integer, ByVal Port As Integer, ByVal Line As Integer, State As Integer) As Integer

**Parameters** **CardNumber:** The card id of the card to perform this operation.

**Port:** Digital output port number. Refer to function [KDAQ\\_DI\\_ReadLine](#) for valid values.

**Line:** The digital line to be accessed. Refer to function [KDAQ\\_DI\\_ReadLine](#) for valid values.

**State:** Returns the digital logic state, 0 or 1, of the specified line.

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel

## KDAQ\_DO\_ReadPort

**Description** Read back the output digital data from the specified digital output port.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

I16 KDAQ\_DO\_ReadPort (U16 CardNumber, U16 Port, U32 \*Value)

### Visual Basic

KDAQ\_DO\_ReadPort (ByVal CardNumber As Integer, ByVal Port As Integer, Value As Long) As Integer

**Parameters** **CardNumber:** The card id of the card to perform this operation.

**Port:** Digital output port number. Refer to function [KDAQ\\_DI\\_ReadPort](#) for valid values.

**Value:** Returns the digital data read from the specified output port. Refer to function [KDAQ\\_DI\\_ReadPort](#) for valid values.

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel

## KDAQ\_DO\_WriteLine

**Description** Sets the specified digital output line in the specified digital port to the specified state. This function is only available for these cards that support digital output read-back functionality.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_DO_WriteLine (U16 CardNumber, U16 Port, U16 Line, U16 State)
```

### Visual Basic

```
KDAQ_DO_WriteLine(ByVal CardNumber As Integer,
    ByVal Port As Integer, ByVal DoLine As Integer,
    ByVal State As Integer) As Integer
```

**Parameters** **CardNumber:** The card id of the card to perform this operation.  
**Port:** Digital output port number. Refer to function [KDAQ\\_DI\\_ReadPort](#) for valid values.  
**Line:** The digital line to write. Refer to function [KDAQ\\_DI\\_ReadPort](#) for valid values.  
**State:** The new digital logic state, 0 or 1.

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel

## KDAQ\_DO\_WritePort

**Description** Writes digital data to the specified digital output port.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_DO_WritePort (U16 CardNumber, U16 Port, U32 Value)
```

### Visual Basic

```
KDAQ_DO_WritePort (ByVal CardNumber As Integer,
    ByVal Port As Integer, ByVal Value As Long) As Integer
```

**Parameters** **CardNumber:** The card id of the card to perform this operation.  
**Port:** Digital output port number. The cards that support this function and their corresponding valid value are as follows:  
 Channel\_P1A, Channel\_P1B,  
 Channel\_P1C, Channel\_P1CL,  
 Channel\_P1CH

**Value:** Digital data that is written to the specified port. The value is 8-bit data.

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel

## KDAQ\_EEPROM\_CAL\_Constant\_Update

- Description** Save new calibration constants to the specified *bank* of EEPROM.
- This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M, KPXI-AO-4-1M, KPXI-AO-8-1M
- Syntax** **Microsoft C/C++ and Borland C++**
- ```
I16 KDAQ_EEPROM_CAL_Constant_Update(U16 wCardNumber, U16 bank)
```
- Visual Basic**
- ```
KDAQ_EEPROM_CAL_Constant_Update (ByVal wCardNumber As Integer, ByVal bank As Integer) As Integer
```
- Parameters** **CardNumber:** The card id of the card to perform this operation.
- bank:** The storage location on EEPROM. The valid range of the value of bank is 0 through 3.
- Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel

## KDAQ\_GCTR\_Control

- Description** Controls for the selected counter/timer by software.
- This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-AO-4-1M, KPXI-AO-8-1M
- Syntax** **Microsoft C/C++ and Borland C++**
- ```
I16 KDAQ_GCTR_Control (U16 wCardNumber, U16 wGctr, U16 ParamID, U16 Value)
```
- Visual Basic**
- ```
KDAQ_GCTR_Control (ByVal CardNumber As Integer, ByVal wGctr As Integer, ByVal ParamID As Integer, ByVal Value As Integer) As Integer
```
- Parameters** **CardNumber:** The card id of the card to perform this operation.
- Gctr:** The counter number. Range: 0 – 1
- ParamID:** The ID of the internal Parameter the general-purpose timer/counter wishes to control. The valid control parameters are as follows:
- GPTC\_IntGATE : Internal gate  
 GPTC\_IntUpDnCTR: internal updown counter  
 GPTC\_IntENABLE:: start or stop counter operation
- Value:** The value for the control item specified by the parameter *ParamID*. The valid value for the are 0 or 1.
- Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, InvalidCounter



### KDAQ\_GCTR\_Read

**Description** Reads the counter value of the general-purpose counter without disturbing the counting process.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_GCTR_Read (U16 wCardNumber, U16 wGCtr, U32 *pValue)
```

**Visual Basic**

```
KDAQ_GCTR_Read (ByVal CardNumber As Integer,
    ByVal GCtr As Integer, Value As Long) As Integer
```

**Parameters** **CardNumber:** The card id of the card to perform this operation.

**GCtr:** The counter number. Range: 0 – 1

**Value:** Returns the counter value of the specified general-purpose timer/counter. Range: 0 – 65535

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, InvalidCounter

### KDAQ\_GCTR\_Reset

**Description** Halts the specified general-purpose timer/counter operation and reloads the initial value of the timer/counter.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_GCTR_Reset (U16 CardNumber, U16 GCtr)
```

**Visual Basic**

```
KDAQ_GCTR_Reset (ByVal CardNumber As Integer,
    ByVal GCtr As Integer) As Integer
```

**Parameters** **CardNumber:** The card id of the card to perform this operation.

**GCtr:** The counter number. Range: 0 – 1

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, InvalidCounter

### KDAQ\_GCTR\_Setup

**Description** Controls the operation of the selected counter/timer.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_GCTR_Setup (U16 wCardNumber, U16 wGCtr, U16 wMode,
    U8 SrcCtrl, U8 PolCtrl, U16 LReg1_Val, U16 LReg2_Val)
```

**Visual Basic**

```
KDAQ_GCTR_Setup (ByVal CardNumber As Integer,
    ByVal wGctr As Integer, ByVal wMode As Integer,
    ByVal SrcCtrl As Byte, ByVal PolCtrl As Byte,
    ByVal LReg1_Val As Integer,
    ByVal LReg2_Val As Integer) As Integer
```

**Parameters** **CardNumber:** The card id of the card to perform this operation.

**Gctr:** The counter number. Range: 0 – 1

**Mode:** The Timer/Counter mode. The valid mode:

```
SimpleGatedEventCNT
SinglePeriodMSR
SinglePulseWidthMSR
SingleGatedPulseGen
SingleTrigPulseGen
RetrigSinglePulseGen
SingleTrigContPulseGen
ContGatedPulseGen
```

Refer to [Section A](#) for additional information regarding mode.

**SrcCtrl:** The setting for general-purpose timer/counter source control. This argument is an integer expression formed from one or more of the manifest constants defined in kdaqdrv.h. There are three groups of constants:

**(1) Timer/Counter Source**

```
GPTC_CLKSRC_INT : internal time base
GPTC_CLKSRC_EXT : external time base from GPTC0_SRC or GPTC1_SRC
pin
```

**(2) Timer/Counter Gate Source**

```
GPTC_GATESRC_INT : gate is controlled by software
GPTC_GATESRC_EXT: gate is controlled by GPTC0_GATE or GPTC1_GATE
pin
```

**(3) Timer/Counter UpDown Source**

```
GPTC_UPDOWN_SEL_INT : Up/Down controlled by software
GPTC_UPDOWN_SEL_EXT: Up/Down controlled by GPTC0_UPDOWN or
GPTC1_UPDOWN pin
```

When two or more constants are used to form the *GCtrl* argument, the constants are combined with the bitwise-OR operator(`|`).

***PolCtrl***: the polarity settings for general-purpose timer/counter

This argument is an integer expression formed from one or more of the manifest constants defined in kdaqdrvr.h. There are four groups of constants:

**(1) Timer/Counter Gate Polarity**

GPTC\_GATE\_LACTIVE : Low active  
 GPTC\_GATE\_HACTIVE : High active

**(2) Timer/Counter UpDown Polarity**

GPTC\_UPDOWN\_LACTIVE : Low active  
 GPTC\_UPDOWN\_HACTIVE: High active

**(3) Timer/Counter ClockEn Polarity**

GPTC\_CLKEN\_LACTIVE : Low active  
 GPTC\_CLKEN\_HACTIVE : High active

**(4) Timer/Counter Output Polarity**

GPTC\_OUTPUT\_LACTIVE : Low active  
 GPTC\_OUTPUT\_HACTIVE: High active

When two or more constants are used to form the *GCtrl* argument, the constants are combined with the bitwise-OR operator(|).

***LReg1\_Val***: The counter value of load register 1 of timer/counter. The meaning for the value depends on the mode the timer/counter performs. For mode 1 to mode 3, the value of *LReg1\_Val* is the initial count of the GPTC. For mode 4 to mode 8 (the pulse generation modes), the value of *LReg1\_Val* is configured as the pulse delay.

***LReg2\_Val***: The counter value of load register 2 of timer/counter. For mode 1 to mode 3, the value of *LReg2\_Val* is not used. For mode 4 to mode 8 (the pulse generation modes), the value of *LReg2\_Val* is configured as the pulse width.

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, InvalidCounter

**KDAQ\_GCTR\_Status**

**Description** Reads the latched GPTC status of the general-purpose counter from the GPTC status register.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_GCTR_Status (U16 wCardNumber, U16 wGCtrl,
    U16 *pValue)
```

**Visual Basic**

```
KDAQ_GCTR_Status (ByVal CardNumber As Integer, ByVal GCtrl As Integer, Value As Integer) As Integer
```

**Parameters**    **CardNumber:** The card id of the card to perform this operation.  
**GCtr:** The counter number. Range: 0 – 1

**Value:** Returns the latched GPTC status of the specified general-purpose timer/counter from the GPTC status register. The format of Value is as follows:

bit 0: formerly latched status of enable  
bit 1: formerly latched status of gate  
bit 2: formerly latched status of up/down  
bit 3: formerly latched status of output  
bit 4: formerly latched status of clk  
bit 5: formerly latched status of interrupt  
bit 6 through 15: not used

**Return Value**    NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, InvalidCounter

### KDAQ\_Load\_CAL\_Data

**Description**    Load calibration constants from the specified *bank* of EEPROM.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax**    **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_Load_CAL_Data (U16 CardNumber, U16 bank)
```

**Visual Basic**

```
KDAQ_Load_CAL_Data (ByVal CardNumber As Integer, ByVal bank As Integer) As Integer
```

**Parameters**    **CardNumber:** The card id of the card to perform this operation.  
**bank:** The storage bank on EEPROM. The valid range of the value of bank is 0 – 3.

**Return Value**    NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel

### KDAQ\_Register\_Card

**Description**    Initializes the hardware and software states of a Keithley PXI DAQ data acquisition card, and then returns a numeric card ID that corresponds to the card initialized. KDAQ\_Register\_Card must be called before any other KDAQ-DRVR library functions can be called for that card. The function initializes the card and variables internal to KDAQ-DRVR library. Because Keithley PXI DAQ devices meet the plug-and-play design, the base address (pass-through address) and IRQ level are assigned by system BIOS directly.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax**    **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_Register_Card (U16 CardType, U16 card_num)
```

**Visual Basic**

```
KDAQ_Register_Card (ByVal CardType As Integer,
    ByVal card_num As Integer) As Integer
```

**Parameters** **CardType:** The type of card to be initialized. Keithley will periodically upgrades KDAQ-DRVR to add support for new Keithley PXI DAQ data acquisition cards. Please refer to *Release Notes* for the card types that the current release of KDAQ-DRVR actually supports. Following are the constants defined in kdaqdrv.h that represent the Keithley PXI DAQ devices that KDAQ-DRVR supports:

```
KPXI_SDAQ_4_2M (for KPXI-SDAQ-4-2M)
KPXI_SDAQ_4_500K (for KPXI-SDAQ-4-500K)
KPXI_DAQ_64_3M (for KPXI-DAQ-64-3M)
KPXI_DAQ_64_500K (for KPXI-DAQ-64-500K)
KPXI_DAQ_64_250K (for KPXI-DAQ-64-250K)
KPXI_DAQ_96_3M (for KPXI-DAQ-96-3M)
KPXI_AO_4_1M (for KPXI-AO-4-1M)
KPXI_AO_8_1M (for KPXI-AO-8-1M)
```

**card\_num:** The sequence number of the card with *the same card type* (as defined in argument *CardType*) plugged in the PXI slot. The card sequence number setting is according to the PXI slot sequence in the mainboard. The first card (in the left-most slot) is with card\_num=0. For example, if there are two KPXI-SDAQ-4-2M cards plugged into your PXI chassis, the KPXI-SDAQ-4-2M card in the left-most slot should be registered with card\_num=0, and the other one with card\_num=1.

This function returns a numeric card id for the card initialized. The range of card id is between 0 and 31. If there is any error occurs, it will return negative error code, the possible error codes are listed below:

**Return Value** ErrorTooManyCardRegistered, ErrorUnknownCardType, ErrorOpenDriverFailed, ErrorOpenEventFailed

**KDAQ\_Release\_Card**

**Description** There are at most 32 cards that can be registered simultaneously. This function is used to tell KDAQ-DRVR library that this registered card is not used currently and can be released. This would make room for a new card to register. Also by the end of a program, you need to use this function to release all cards that were registered.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_Release_Card (U16 CardNumber)
```

**Visual Basic**

```
KDAQ_Release_Card (ByVal CardNumber As Integer) As Integer
```

**Parameters** **CardNumber:** The card id of the card to be released.

**Return Value** NoError

## KDAQ\_SSI\_SourceClear

**Description** Disconnects all of the device signals from the SSI bus trigger lines.  
This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_SSI_SourceClear (USHORT wCardNumber)
```

### Visual Basic

```
KDAQ_SSI_SourceClear (ByVal CardNumber As Integer)  
As Integer
```

**Parameters** **CardNumber:** The card id of the card to perform this operation.

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, InvalidCounter

## KDAQ\_SSI\_SourceConn

**Description** Connects a device to the specified SSI bus trigger line.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_SSI_SourceConn (USHORT wCardNumber, USHORT sigCode)
```

### Visual Basic

```
KDAQ_SSI_SourceConn (ByVal CardNumber As Integer,  
ByVal sigCode As Integer) As Integer
```

**Parameters** **CardNumber:** The card id of the card to perform this operation.

**sigCode:** The specified SSI signal code number of the device signal to be connected to the SSI bus trigger line. The direction of the connection is transmitted from the device to the SSI bus trigger line. The valid signal codes are as follows:

```
SSI_TIME : SSI_TIMEBASE output  
SSI_CONV : SSI_ADCONV output  
SSI_WR : SSI_DAWR output  
SSI_ADTRIG : SSI_ADTRIG output  
SSI_DATRIG : SSI_DATRIG output
```

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, InvalidCounter

## KDAQ\_SSI\_SourceDisConn

**Description** Disconnects a device signal from the specified SSI bus trigger line.

This function is supported by the following models: KPXI-SDAQ-4-2M, KPXI-SDAQ-4-500K, KPXI-DAQ-64-3M, KPXI-DAQ-64-500K, KPXI-DAQ-64-250K, KPXI-DAQ-96-3M, KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax**      **Microsoft C/C++ and Borland C++**

```
I16 KDAQ_SSI_SourceDisConn (USHORT wCardNumber,
    USHORT sigCode)
```

**Visual Basic**

```
KDAQ_SSI_SourceDisConn (ByVal CardNumber As Integer,
    ByVal sigCode As Integer) As Integer
```

**Parameters**      **CardNumber:** The card id of the card to perform this operation.

**sigCode:** The specified SSI signal code number of the device signal to be disconnected from the SSI bus trigger line. The valid signal codes are as follows:

- SSI\_TIME : SSI\_TIMEBASE output
- SSI\_CONV : SSI\_ADCONV output
- SSI\_WR : SSI\_DAWR output
- SSI\_ADTRIG : SSI\_ADTRIG output
- SSI\_DATRIG : SSI\_DATRIG output

**Return Value**      NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, InvalidCounter

**SDAQ4K500\_Acquire\_AD\_Error**

**Description**      Acquires the offset and gain errors of the specified AI channel in the specified polarity mode.

This function is supported by the following model: KPXI-SDAQ-4-500K

**Syntax**      **Microsoft C/C++ and Borland C++**

```
I16 SDAQ4K500_Acquire_AD_Error(U16 wCardNumber, U16 channel,
    U16 polarity, float *gain_err, float *offset_err)
```

**Visual Basic**

```
SDAQ4K500_Acquire_AD_Error (ByVal wCardNumber As Integer,
    ByVal channel As Integer, ByVal polarity As Integer,
    gain_err As Single, offset_err As Single) As Integer
```

**Parameters**      **CardNumber:** The card id of the card to perform this operation.

**Channel:** Analog input channel number. Range: 0 – 3

**Polarity:**      The polarity (unipolar or bipolar) of the input channel.  
Valid values: 1: bipolar, 0: unipolar

**gain\_err:** Returns the gain error of the specified AI channel.

**offset\_err:** Returns the offset error of the specified AI channel.

**Return Value**      NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel

## SDAQ4K500\_Acquire\_DA\_Error

**Description** Acquires the offset and gain errors of the specified DA channel in the specified polarity mode.

This function is supported by the following model: KPXI-SDAQ-4-500K

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 SDAQ4K500_Acquire_DA_Error (U16 wCardNumber,
    U16 channel, U16 polarity, float *gain_err,
    float *offset_err)
```

### Visual Basic

```
SDAQ4K500_Acquire_DA_Error (ByVal wCardNumber As Integer,
    ByVal channel As Integer, ByVal polarity As Integer,
    gain_err As Single, offset_err As Single) As Integer
```

**Parameters** **CardNumber:** The card id of the card to perform this operation.

**Channel:** Analog output channel number. Range: 0 – 1

**Polarity:** The polarity (unipolar or bipolar) of the output channel. Valid values: KDAQ\_DA\_BiPolar: bipolar, KDAQ\_DA\_UniPolar: unipolar

**gain\_err:** Returns the gain error of the specified AO channel.

**offset\_err:** Returns the offset error of the specified AO channel.

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel

## SDAQ4M2\_Acquire\_AD\_Error

**Description** Acquires the offset and gain errors of the specified AI channel in the specified polarity mode.

This function is supported by the following model: KPXI-SDAQ-4-2M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 SDAQ4M2_Acquire_AD_Error(U16 wCardNumber, U16 channel,
    U16 polarity, float *gain_err, float *offset_err)
```

### Visual Basic

```
SDAQ4M2_Acquire_AD_Error (ByVal wCardNumber As Integer,
    ByVal channel As Integer, ByVal polarity As Integer,
    gain_err As Single, offset_err As Single) As Integer
```

**Parameters** **CardNumber:** The card id of the card to perform this operation.

**Channel:** Analog input channel number. Range: 0 – 3 for KPXI-SDAQ-4-2M

**Polarity:** The polarity (unipolar or bipolar) of the input channel. Valid values: 1: bipolar, 0: unipolar

**gain\_err:** Returns the gain error of the specified AI channel.

**offset\_err:** Returns the offset error of the specified AI channel.



**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel

### SDAQ4M2\_Acquire\_DA\_Error

**Description** Acquires the offset and gain errors of the specified DA channel in the specified polarity mode.

This function is supported by the following model: KPXI-SDAQ-4-2M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 SDAQ4M2_Acquire_DA_Error(U16 wCardNumber, U16 channel,
    U16 polarity, float *gain_err, float *offset_err)
```

#### Visual Basic

```
SDAQ4M2_Acquire_DA_Error (ByVal wCardNumber As Integer,
    ByVal channel As Integer, ByVal polarity As Integer,
    gain_err As Single, offset_err As Single) As Integer
```

**Parameters** **CardNumber:** The card id of the card to perform this operation.

**Channel:** Analog output channel number. Range: 0 – 1

**Polarity:** The polarity (unipolar or bipolar) of the output channel. Valid values: KDAQ\_DA\_BiPolar: bipolar, KDAQ\_DA\_UniPolar: unipolar

**gain\_err:** Returns the gain error of the specified AO channel.

**offset\_err:** Returns the offset error of the specified AO channel.

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel

### DAQ64M3\_Acquire\_AD\_Error

**Description** Acquire the offset and gain errors of ADC.

This function is supported by the following model: KPXI-DAQ-64-3M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 DAQ64M3_Acquire_AD_Error(U16 wCardNumber,
    float *gain_err, float *bioffset_err,
    float *unioffset_err, float *hg_bios_err)
```

#### Visual Basic

```
DAQ64M3_Acquire_AD_Error (ByVal wCardNumber As Integer,
    gain_err As Single, bioffset_err As Single,
    unioffset_err As Single, hg_bios_err As Single) As Integer
```

**Parameters** **CardNumber:** The card id of the card to perform this operation.

**gain\_err:** Returns the gain error of the ADC.

**bioffset\_err:** Returns the offset error of the ADC in bipolar mode.

**unioffset\_err:** Returns the offset error of the ADC in unipolar mode.

**Hg\_bios\_err:** Returns the high-gain offset error of the ADC in bipolar mode.

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel

### DAQ64M3\_Acquire\_DA\_Error

**Description** Acquire the offset and gain errors of the specified DA channel in the specified polarity.

This function is supported by the following model: KPXI-DAQ-64-3M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 DAQ64M3_Acquire_DA_Error(U16 wCardNumber, U16 channel,
    U16 polarity, float *da0v_err, float *da5v_err)
```

#### Visual Basic

```
DAQ64M3_Acquire_DA_Error (ByVal wCardNumber As Integer,
    ByVal channel As Integer, ByVal polarity As Integer,
    da0v_err As Single, da5v_err As Single) As Integer
```

**Parameters** **CardNumber:** The card id of the card to perform this operation.

**Channel:** Analog output channel number. Range: 0 – 1

**Polarity:** The polarity (unipolar or bipolar) of the output channel. Valid values: KDAQ\_DA\_BiPolar: bipolar, KDAQ\_DA\_UniPolar: unipolar

**Da0v\_err:** Returns the offset error of the specified AO channel.

**Da5v\_err:** Returns the gain error of the specified AO channel.

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel

### DAQ64K500\_Acquire\_AD\_Error

**Description** Acquire the offset and gain errors of ADC.

This function is supported by the following model: KPXI-DAQ-64-500K

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 DAQ64K500_Acquire_AD_Error(U16 wCardNumber,
    float *gain_err, float *bioffset_err,
    float *unioffset_err, float *hg_bios_err)
```

#### Visual Basic

```
DAQ64K500_Acquire_AD_Error (ByVal wCardNumber As Integer,
    gain_err As Single, bioffset_err As Single,
    unioffset_err As Single, hg_bios_err As Single) As Integer
```

**Parameters** **CardNumber:** The card id of the card to perform this operation.

**gain\_err:** Returns the gain error of the ADC.

**bioffset\_err:** Returns the offset error of the ADC in bipolar mode.

**unioffset\_err:** Returns the offset error of the ADC in unipolar mode.

**Hg\_bios\_err:** Returns the high-gain offset error of the ADC in bipolar mode.

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel

**DAQ64K500\_Acquire\_DA\_Error**

**Description** Acquire the offset and gain errors of the specified DA channel in the specified polarity.

This function is supported by the following models: KPXI-DAQ-64-500K

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 DAQ64K500_Acquire_DA_Error(U16 wCardNumber, U16 channel,
    U16 polarity, float *da0v_err, float *da5v_err)
```

**Visual Basic**

```
DAQ64K500_Acquire_DA_Error (ByVal wCardNumber As Integer,
    ByVal channel As Integer, ByVal polarity As Integer,
    da0v_err As Single, da5v_err As Single) As Integer
```

**Parameters** **CardNumber:** The card id of the card to perform this operation.

**Channel:** Analog output channel number.

Range: 0 through 1

**Polarity:** The polarity (unipolar or bipolar) of the output channel. Valid values: KDAQ\_DA\_BiPolar: bipolar, KDAQ\_DA\_UniPolar: unipolar

**Da0v\_err:** Returns the offset error of the specified AO channel.

**Da5v\_err:** Returns the gain error of the specified AO channel.

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel

**DAQ64K250\_Acquire\_AD\_Error**

**Description** Acquire the offset and gain errors of ADC.

This function is supported by the following model: KPXI-DAQ-64-250K

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 DAQ64K250_Acquire_AD_Error(U16 wCardNumber,
    float *gain_err, float *bioffset_err,
    float *unioffset_err, float *hg_bios_err)
```

**Visual Basic**

```
DAQ64K250_Acquire_AD_Error (ByVal wCardNumber As Integer,
    gain_err As Single, bioffset_err As Single,
    unioffset_err As Single, hg_bios_err As Single) As Integer
```

**Parameters** *CardNumber*: The card id of the card to perform this operation.  
*gain\_err*: Returns the gain error of the ADC.  
*bioffset\_err*: Returns the offset error of the ADC in bipolar mode.  
*unioffset\_err*: Returns the offset error of the ADC in unipolar mode.  
*Hg\_bios\_err*: Returns the high-gain offset error of the ADC in bipolar mode.

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel

### DAQ64K250\_Acquire\_DA\_Error

**Description** Acquire the offset and gain errors of the specified DA channel in the specified polarity.

This function is supported by the following model: KPXI-DAQ-64-250K

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 DAQ64K250_Acquire_DA_Error(U16 wCardNumber, U16 channel,
    U16 polarity, float *da0v_err, float *da5v_err)
```

#### Visual Basic

```
DAQ64K250_Acquire_DA_Error (ByVal wCardNumber As Integer,
    ByVal channel As Integer, ByVal polarity As Integer,
    da0v_err As Single, da5v_err As Single) As Integer
```

**Parameters** *CardNumber*: The card id of the card to perform this operation.  
*Channel*: Analog output channel number. Range: 0 through 1  
*Polarity*: The polarity (unipolar or bipolar) of the output channel. Valid values: KDAQ\_DA\_BiPolar: bipolar, KDAQ\_DA\_UniPolar: unipolar  
*Da0v\_err*: Returns the offset error of the specified AO channel.  
*Da5v\_err*: Returns the gain error of the specified AO channel.

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel

### DAQ96M3\_Acquire\_AD\_Error

**Description** Acquire the offset and gain errors of ADC.

This function is supported by the following model: KPXI-DAQ-96-3M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 DAQ96M3_Acquire_AD_Error(U16 wCardNumber,
    float *gain_err, float *bioffset_err,
    float *unioffset_err, float *hg_bios_err)
```

#### Visual Basic

```
DAQ96M3_Acquire_AD_Error (ByVal wCardNumber As Integer,
    gain_err As Single, bioffset_err As Single,
    unioffset_err As Single, hg_bios_err As Single) As Integer
```

**Parameters** *CardNumber*: The card id of the card to perform this operation.  
*gain\_err*: Returns the gain error of the ADC.  
*bioffset\_err*: Returns the offset error of the ADC in bipolar mode.  
*unioffset\_err*: Returns the offset error of the ADC in unipolar mode.  
*Hg\_bios\_err*: Returns the high-gain offset error of the ADC in bipolar mode.

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel

### AOxM1\_Acquire\_AD\_Error

**Description** Acquires the offset and gain errors in the specified polarity mode.  
 This function is supported by the following models: KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 AOxM1_Acquire_AD_Error(I16 wCardNumber, U16 polarity,
    float *gain_err, float *offset_err)
```

**Visual Basic**

```
AOxM1_Acquire_AD_Error (ByVal wCardNumber As Integer,
    ByVal polarity As Integer, gain_err As Single,
    offset_err As Single) As Integer
```

**Parameters** *CardNumber*: The card id of the card to perform this operation.  
*Polarity*: The polarity (unipolar or bipolar) of the input channel.  
 Valid values : 1 (bipolar), 0 (unipolar)  
*gain\_err*: Returns the gain error of the specified AI channel.  
*offset\_err*: Returns the offset error of the specified AI channel.

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport

### AOxM1\_Acquire\_DA\_Error

**Description** Acquires the offset and gain errors of the specified DA channel in the specified polarity mode.  
 This function is supported by the following models: KPXI-AO-4-1M, KPXI-AO-8-1M

**Syntax** **Microsoft C/C++ and Borland C++**

```
I16 AOxM1_Acquire_DA_Error(U16 wCardNumber, U16 channel,
    U16 polarity, float *gain_err, float *offset_err)
```

**Visual Basic**

```
AOxM1_Acquire_DA_Error (ByVal wCardNumber As Integer,
    ByVal channel As Integer, ByVal polarity As Integer,
    gain_err As Single, offset_err As Single) As Integer
```

**Parameters** *CardNumber*: The card id of the card to perform this operation.  
*Channel*: Analog output channel number.

Range:  
 0–3 for KPXI-AO-4-1M  
 0–7 for KPXI-AO-8-1M

**Polarity:** The polarity (unipolar or bipolar) of the output channel. Valid values:

KDAQ\_DA\_BiPolar: bipolar

KDAQ\_DA\_UniPolar: unipolar

**gain\_err:** Returns the gain error of the specified AO channel.

**offset\_err:** Returns the offset error of the specified AO channel.

**Return Value** NoError, ErrorInvalidCardNumber, ErrorCardNotRegistered, ErrorFuncNotSupport, ErrorInvalidIoChannel

## Status Codes

This paragraph lists status codes returned by KDAQ-DRVR (including the name and description).

Each KDAQ-DRVR function returns a status code that indicates whether the function was performed successfully. When a KDAQ-DRVR function returns a negative number, it means that an error occurred while executing the function.

Table B-3

**Status codes returned by KDAQ-DRVR**

Code	Status Name	Description
0	NoError	No error occurred
-1	ErrorUnknownCardType	The <i>CardType</i> argument is not valid
-2	ErrorInvalidCardNumber	The <i>CardNumber</i> argument is out of range (larger than 31).
-3	ErrorTooManyCardRegistered	There have been 32 cards that were registered.
-4	ErrorCardNotRegistered	No card registered as id <i>CardNumber</i> .
-5	ErrorFuncNotSupport	The function called is not supported by this type of card..
-6	ErrorInvalidIoChannel	The specified <i>Channel</i> or <i>Port</i> argument is out of range..
-7	ErrorInvalidAdRange	The specified analog input range is invalid.
-8	ErrorContIoNotAllowed	The specified continuous IO operation is not supported by this type of card.
-9	ErrorDiffRangeNotSupport	All the analog input ranges must be the same for multi-channel analog input.
-10	ErrorLastChannelNotZero	The channels for multi-channel analog input must be ended with or started from zero.
-11	ErrorChannelNotDescending	The channels for multi-channel analog input must be contiguous and in descending order.
-12	ErrorChannelNotAscending	The channels for multi-channel analog input must be contiguous and in ascending order.
-13	ErrorOpenDriverFailed	Failed to open the device driver.
-14	ErrorOpenEventFailed	Open event failed in device driver.
-15	ErrorTransferCountTooLarge	The size of transfer is larger than the size of Initially allocated memory in driver.
-16	ErrorNotDoubleBufferMode	Double buffer mode is disabled.
-17	ErrorInvalidSampleRate	The specified sampling rate is out of range.
-18	ErrorInvalidCounterMode	The value of the <i>Mode</i> argument is invalid.
-19	ErrorInvalidCounter	The value of the <i>Ctr</i> argument is out of range.
-20	ErrorInvalidCounterState	The value of the <i>State</i> argument is out of range.
-21	ErrorInvalidBinBcdParam	The value of the <i>BinBcd</i> argument is invalid.
-22	ErrorBadCardType	The value of Card Type argument is invalid
-23	ErrorInvalidDaRefVoltage	The value of DA reference voltage argument is invalid

Table B-3 (continued)  
**Status codes returned by KDAQ-DRVR**

Code	Status Name	Description
-24	ErrorAdTimeOut	Time out for AD operation
-25	ErrorNoAsyncAI	Continuous Analog Input is not set as Asynchronous mode
-26	ErrorNoAsyncAO	Continuous Analog Output is not set as Asynchronous mode
-27	ErrorNoAsyncDI	Continuous Digital Input is not set as Asynchronous mode
-28	ErrorNoAsyncDO	Continuous Digital Output is not set as Asynchronous mode
-29	ErrorNotInputPort	The value of AI/DI port argument is invalid
-30	ErrorNotOutputPort	The value of AO/DO argument is invalid
-31	ErrorInvalidDioPort	The value of DI/O port argument is invalid
-32	ErrorInvalidDioLine	The value of DI/O line argument is invalid
-33	ErrorContIoActive	Continuous IO operation is not active
-34	ErrorDbIBufModeNotAllowed	Double Buffer mode is not allowed
-35	ErrorConfigFailed	The specified function configuration is failed
-36	ErrorInvalidPortDirection	The value of DIO port direction argument is invalid
-37	ErrorBeginThreadError	Failed to create thread
-38	ErrorInvalidPortWidth	The port width setting is not allowed
-39	ErrorInvalidCtrSource	The clock source setting is invalid
-40	ErrorOpenFile	Failed to Open file
-41	ErrorAllocateMemory	The memory allocation is failed
-42	ErrorDaVoltageOutOfRange	The value of DA voltage argument is out of range
-43	ErrorInvalidSyncMode	The sync. mode of operation is invalid
-44	ErrorInvalidBufferID	The buffer id selected is invalid
-45	ErrorInvalidCNTInterval	The counter value is invalid
-46	ErrorReTrigModeNotAllowed	The Re-Trigger mode of operation is invalid
-47	ErrorResetBufferNotAllowed	The buffer is not allowed to be reset
-48	ErrorAnaTriggerLevel	The value of analog trigger level is invalid
-49	ErrorDAQEvent	The DAQEvent is invalid
-201	ErrorConfigIoctl	The configuration API is failed
-202	ErrorAsyncSetIoctl	The async. mode API is failed
-203	ErrorDBSetIoctl	The double-buffer setting API is failed
-204	ErrorDBHalfReadyIoctl	The half-ready API is failed
-205	ErrorContOPIoctl	The continuous data acquisition API is failed
-206	ErrorContStatusIoctl	The continuous data acquisition status API setting is failed
-207	ErrorPIIoctl	The polling data API is failed
-208	ErrorDIntSetIoctl	The dual interrupt setting API is failed
-209	ErrorWaitEvtIoctl	The wait event API is failed
-210	ErrorOpenEvtIoctl	The open event API is failed
-211	ErrorCOSIntSetIoctl	The COS interrupt setting API is failed
-212	ErrorMemMapIoctl	The memory mapping API is failed
-213	ErrorMemUMapSetIoctl	The memory Un-mapping API is failed
-214	ErrorCTRIoctl	The counter API is failed
-215	ErrorGetResIoctl	The resource getting API is failed

## AI range codes

The analog input range for Keithley Instruments PXI KDAQ modules is contained in [Table B-4](#) with valid values in [Table B-4](#).

Table B-4  
Analog input range of digitizers

Input	Range
AD_B_10_V	Bipolar -10V to +10V
AD_B_5_V	Bipolar -5V to +5V
AD_B_2_5_V	Bipolar -2.5V to +2.5V
AD_B_1_25_V	Bipolar -1.25V to +1.25V
AD_B_0_625_V	Bipolar -0.625V to +0.625V
AD_B_0_3125_V	Bipolar -0.3125V to +0.3125V
AD_B_0_5_V	Bipolar -0.5V to +0.5V
AD_B_0_05_V	Bipolar -0.05V to +0.05V
AD_B_0_005_V	Bipolar -0.005V to +0.005V
AD_B_1_V	Bipolar -1V to +1V
AD_B_0_1_V	Bipolar -0.1V to +0.1V
AD_B_0_01_V	Bipolar -0.01V to +0.01V
AD_B_0_001_V	Bipolar -0.01V to +0.001V
AD_U_20_V	Unipolar 0 to +20V
AD_U_10_V	Unipolar 0 to +10V
AD_U_5_V	Unipolar 0 to +5V
AD_U_2_5_V	Unipolar 0 to +2.5V
AD_U_1_25_V	Unipolar 0 to +1.25V
AD_U_1_V	Unipolar 0 to +1V
AD_U_0_1_V	Unipolar 0 to +0.1V
AD_U_0_01_V	Unipolar 0 to +0.01V
AD_U_0_001_V	Unipolar 0 to +0.001V
AD_B_2_V	Bipolar -2V to +2V
AD_B_0_25_V	Bipolar -0.25V to +0.25V
AD_B_0_2_V	Bipolar -0.2V to +0.2V
AD_U_4_V	Unipolar 0 to +4V
AD_U_2_V	Unipolar 0 to +2V
AD_U_0_5_V	Unipolar 0 to +0.5V
AD_U_0_4_V	Unipolar 0 to +0.4V

Table B-5  
Valid values for each model

Model	Valid value
KPXI-SDAQ-4-500K KPXI-SDAQ-4-2M KPXI-DAQ-64-500K KPXI-DAQ-64-250K	AD_B_10_V, AD_B_5_V, AD_B_2_5_V, AD_B_1_25_V AD_U_10_V, AD_U_5_V, AD_U_2_5_V, AD_U_1_25_V
KPXI-DAQ-64-3M KPXI-DAQ-96-3M	AD_B_10_V, AD_B_5_V, AD_B_2_5_V, AD_B_2_V, AD_B_1_25_V, AD_B_1_V, AD_B_0_5_V, AD_B_0_25_V, AD_B_0_2_V, AD_B_0_05_V, AD_U_10_V, AD_U_5_V, AD_U_4_V, AD_U_2_5_V, AD_U_2_V, AD_U_1_V, AD_U_0_5_V, AD_U_0_4_V, AD_U_0_1_V
KPXI-AO-4-1M KPXI-AO-8-1M	AD_B_10_V, AD_U_10_V



## AI data format

Table B-6 lists the AI data format for the cards performing analog input operation, as well as the calculation methods to retrieve the A/D converted data and the channel from where the data was read.

Table B-6  
AI data format

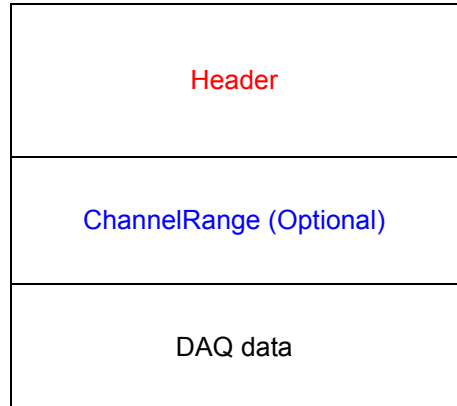
Card Type	Data Format	Value calculation * channel no. (CH#) * A/D converted data (ND) * Value returned from AI function (OD)
KPXI-SDAQ-4-2M	Every 16-bit signed integer data: D13 D12 ..... D1D0 b1 b0  where: D13,D12, ... , D0: A/D converted data b1, b0: Simultaneous Digital Input data.	ND = OD >>2 or ND = OD/4
KPXI-SDAQ-4-500K	Every 16-bit signed integer data: D15 D14 D13..... D1D0  where: D15,D14, ... , D0: A/D converted data	ND = OD
KPXI-DAQ-64-3M	Every 16-bit signed integer data: D12 D11 ..... D1D0 b3 b2 b1 b0  where: D12,D11, ... , D0: A/D converted data b3, b2, b1, b0: Simultaneous Digital Input data.	ND = OD >>4 or ND = OD/16
KPXI-DAQ-64-500K KPXI-DAQ-64-250K	Every 16-bit signed integer data: D15 D14 D13..... D1D0  where: D15,D14, ... , D0: A/D converted data	ND = OD
KPXI-DAQ-96-3M	Every 16-bit signed integer data: D12 D11 ..... D1D0 b3 b2 b1 b0  where: D12,D11, ... , D0: A/D converted data b3, b2, b1, b0: not used.	ND = OD >>4 or ND = OD/16
KPXI-AO-4-1M KPXI-AO-8-1M	Every 16-bit signed integer data: D13 D12 ..... D1D0 b1 b0  where: D13,D12, ... , D0 : A/D converted data b1, b0 : AI Auto-scan Channel.	ND = OD >>2 or ND = OD/4

## DATA file format

This paragraph describes the file format of the data files generated by the functions performing continuous data acquisition followed by storing the data to disk.

The data file includes three parts, Header, ChannelRange (optional) and Data block. The file structure is as is [Figure B-9](#).

Figure B-9  
DATA file format



## Header

The *header* part records the information related to the stored data and its total length is 60 bytes. The data structure of the file header is contained in [Table B-7](#).

Table B-7  
Data file header

Header			<i>Total Length: 60 bytes</i>
Elements	Type	Size (bytes)	Comments
ID	char	10	file ID ex. KeithleyDAQ2
card_type	short	2	card Type ex. KPXI_SDAQ-4-2M
num_of_channel	short	2	number of scanned channels ex. 1, 2
Channel_no	unsigned char	1	channel number where the data read from (only available as the num_of_channel is 1) ex. 0, 1
num_of_scan	long	4	the number of scan for each channel (total count / num_of_channel)
data_width	short	2	the data width 0: 8 bits, 1: 16 bits, 2: 32 bits
channel_order	short	2	the channel scanned sequence 0: normal (ex. 0-1-2-3) 1: reverse (ex. 3-2-1-0) 2: custom* (ex. 0, 1, 3)
ad_range	short	2	the AI range code Please refer to <a href="#">DATA file format</a> ex. 0 (AD_B_10V)
scan_rate	double	8	The scanning rate of each channel (total sampling rate / num_of_channel)
num_of_channel_range	short	2	The number of ChannelRange* structure
start_date	char	8	The starting date of data acquisition ex. 11/11/06

\* If the num\_of\_channel\_range is 0, the ChannelRange block won't be included in the data file.  
\* The channel\_order is set to "custom" only when the card supports variant channel scanning order.

Table B-7 (continued)

**Data file header**

Header			Total Length: 60 bytes
Elements	Type	Size (bytes)	Comments
start_time	char	8	The starting time of data acquisition ex. 18:30:25
start_millisecond	char	3	The starting millisecond of data acquisition ex. 360
reserved	char	6	not used

\* If the num\_of\_channel\_range is 0, the ChannelRange block won't be included in the data file.  
 \* The channel\_order is set to "custom" only when the card supports variant channel scanning order.

**ChannelRange**

The *ChannelRange* part records the channel number and data range information related to the stored data. This part consists of several channel and range units. The length of each unit is 2 bytes. The total length depends on the value of *num\_of\_channel\_range* (one element of the file header) and is calculated as the following formula:

$$\text{Total Length} = 2 * \text{num\_of\_channel\_range bytes}$$

The data structure of each ChannelRange unit is contained in [Table B-8](#):

Table B-8

**Data structure of ChannelRange unit (length: 2 bytes)**

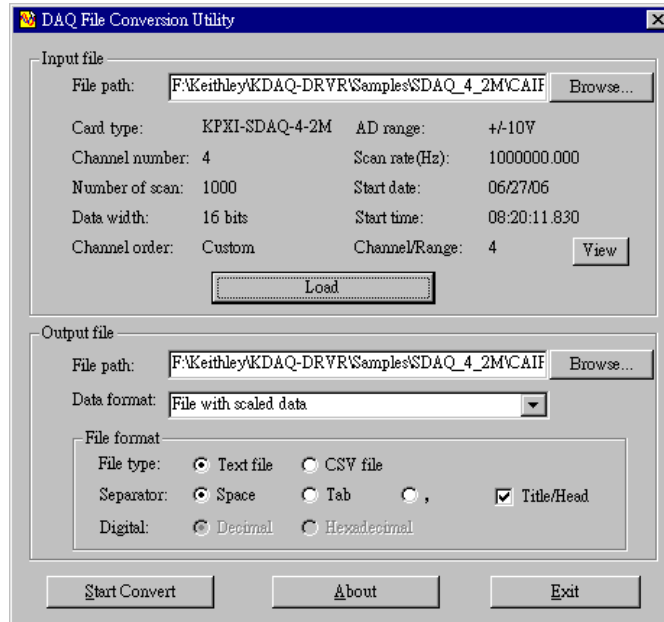
Elements	Type	Size (bytes)	Comments
channel	char	1	scanned channel number ex. 0, 1
range	char	1	the AI range code of <i>channel</i> Please refer to <a href="#">DATA file format</a> . ex. 0 (AD_B_10V)

**Data Block**

The last part is the data block. The data is written to file in 16-bit binary format, with the lower byte first (little endian). For example, the value 0x1234 is written to disk with 34 first followed by 12. The total length of the data block depends on the data width and the total data count.

The file is written in Binary format and can't be read in normal text editor. You can use any binary file editor to view it or the functions used for reading files, e.g. `fread`, to get the file information and data value. KDIO-DRVR provides a useful utility *KiDAQCvt* for you to convert the binary file. The *KiDAQCvt* main window is shown in [Figure B-10](#).

Figure B-10  
**DAQ File Conversion Utility**



KiDAQCvt first translates the information stored in the header part and the ChannelRange part and then displays the corresponding information in the “Input File” frame of *KiDAQCvt* main window. After setting the properties (File Path, Format, ...etc.) of the converted file and push “*Start Convert*” button in the “Output File” frame, *KiDAQCvt* gets rid of header and ChannelRange parts and converts the data in data block according to the card type and the data width. Finally, *KiDAQCvt* writes the converted data to disk. You thus can use any text editor or Excel to view or analyze the accessed data.

---

# KIDAQ<sup>®</sup>-LabVIEW Compatible Interface Guide

## In this appendix:

Topic	Page
<b>Introduction to KIDAQ<sup>®</sup>-LabVIEW</b> .....	C-2
Overview .....	C-2
Using KIDAQ LabVIEW VIs in LabVIEW .....	C-2
KIDAQ LabVIEW Programming .....	C-3
<b>Device Driver Handling</b> .....	C-4
Windows XP/2000 Device Driver .....	C-4
Driver Utility .....	C-4
<b>KIDAQ Utilities</b> .....	C-4
KIDAQ Registry/Configuration utility .....	C-4
KIDAQ Device Browser .....	C-4
<b>KIDAQ LabVIEW VIs Overview</b> .....	C-5
Analog Input VIs .....	C-6
Analog Output VIs .....	C-6
Digital I/O VIs .....	C-7
Timer/Counter VIs .....	C-7
Calibration and Configuration VIs .....	C-8
Error Handler VI .....	C-8
<b>Distribution of Applications</b> .....	C-8
Windows XP/2000 .....	C-8

## Introduction to KIDAQ®-LabVIEW

This introduction describes how to program your application in LabVIEW<sup>1</sup> using the Keithley KIDAQ driver.

### Overview

Install the KDAQ-DRV, KDIO-DRV, or KDIG-DRV device driver that works with your module before installing the KIDAQ LabVIEW driver. Refer to driver installation information elsewhere in the product manual for the correct driver installation procedure for your module.

KIDAQ LabVIEW VIs (Virtual Instrumentation files) were designed for LabVIEW 6.0 or later. All VIs are stored in 6.0 format. The KIDAQ driver provides a set of VIs that control the KPXI modules from within LabVIEW for fast and simple programming.

To not conflict with the naming of the functions already present in LabVIEW, all KIDAQ LabVIEW VIs have a “KI” prefix. For example, the Analog Input Read VI is called “KI AI Read”.

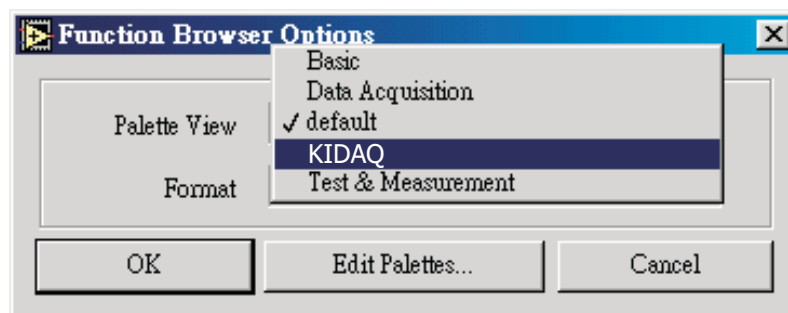
### Using KIDAQ LabVIEW VIs in LabVIEW

To use KIDAQ LabVIEW VIs, refer to the following procedure as a guideline (using LabVIEW versions 6.0 through 7.2):

**NOTE** *LabVIEW 8 (and later versions) uses a new interface. In LabVIEW 8, the KIDAQ VI set will appear at the bottom of the LabVIEW function palette. To personalize your function palette, click the **Tools menu** item, select **Advanced and edit palette set...** from the menus.*

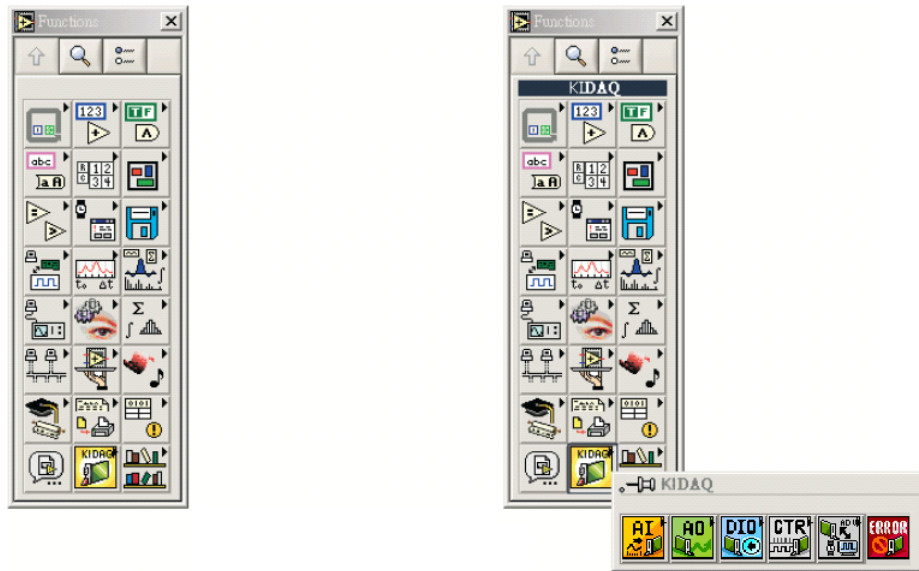
1. Click the **Options** button in the **Controls** or **Functions** palette toolbar to display the **Function Browser Options** dialog box.
2. Select **KIDAQ** view from the **Palette Set** pull-down menu (Figure C-1).
3. Click **OK**. The **Functions** palettes change to the **KIDAQ** view.
4. Then, find KIDAQ LabVIEW VIs in **KIDAQ** icon on the **Functions** palette (Figure C-2).

Figure C-1  
Function Browser Options



1. LabVIEW™ is a trademark of the National Instruments Corporation. All other trademarks are the property of their respective owners.

Figure C-2  
**Functions palette**



The **KIDAQ** palette contains four sub-palettes that contain the different classes of data acquisition VIs. The VIs are classified as follows:

- Analog Input VIs
- Analog Output VIs
- Digital I/O VIs
- Timer/Counter VIs
- Calibration and Configuration VIs
- Error Handler VI

Most of the VI sub-palettes arrange the VIs in different levels, Easy, Intermediate, or Advanced, according to their functionality.

## KIDAQ LabVIEW Programming

The [KIDAQ LabVIEW VIs Overview](#) briefly describes each VI in KIDAQ LabVIEW. All applications developed with KIDAQ LabVIEW are compatible across Windows XP and 2000. For detailed function information, refer to [Appendix D](#), the [KIDAQ®-LabVIEW Compatible Function Reference](#).

You can find the detailed description of each VI using any of the following ways:

- Select the **Show Help** command in the **Help** menu in LabVIEW. Then, when you put the mouse cursor on KIDAQ LabVIEW VI, LabVIEW will show the description of the VI.
- Refer to [Appendix D](#) of this [User's Manual](#).
- Contact Keithley Instruments via phone, email, or on the web at [www.keithley.com](http://www.keithley.com) for further information.

## Device Driver Handling

Device Driver Handling describes how to configure the KIDAQ PXI cards Windows® XP/2000 device driver.

### Windows XP/2000 Device Driver

Once Windows XP/2000® has started, the Plug and Play function of Windows XP/2000® operating system will find the new Keithley PXI cards. If this is the first time to install Keithley PXI cards in your Windows XP/2000® system, you will be informed to install the device driver. Refer to driver installation information elsewhere in the product manual for the correct driver installation procedure for your module.

### Driver Utility

**NOTE** *The KDAQ-DRVR, KDIO-DRVR, or KDIG-DRVR device driver should be installed before the KDAQ LabVIEW driver. Refer to driver installation information elsewhere in the product manual for the correct driver installation procedure for your module.*

KIDAQ LabVIEW provides a PXI Configuration Utility (*configdrv.exe*). These utilities are used to **set/change** the allocated buffer sizes of AI, AO, DI and DO (Analog Input, Analog Output, Digital Input, Digital Output). The allocated buffer sizes of AI, AO, DI, DO represent the sizes of contiguous Initially Allocated memory for continuous analog input, analog output, digital input, digital output respectively. Its unit is page *KB*, i.e. 1024 bytes. The device driver will try to allocate these sizes of memory at system startup time. If this size of memory is not available, the driver will allocate as much memory as system can provide. The size of initially allocated memory is the maximum memory size that DMA or Interrupt transfer can be performed. It will induce an unexpected result in that DMA or Interrupt transfer performed exceeds the initially allocated size.

## KIDAQ Utilities

This section, KIDAQ Utilities, describes all utilities included in the KIDAQ software.

### KIDAQ Registry/Configuration utility

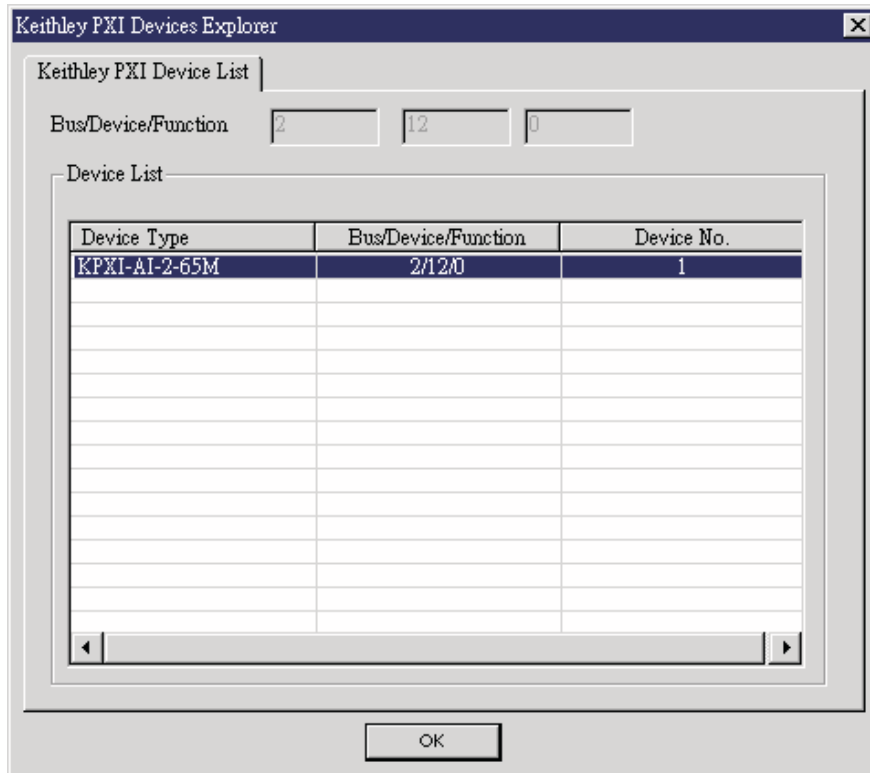
*configdrv* is used to modify the allocated buffer sizes of AI, AO, DI and DO (Windows® XP/2000). The default installation directory for this utility is **C:\Keithley\KIDAQUtil**. It can also be found in the start menu under **Programs -> Keithley -> KIDAQ LabVIEW Driver -> Configuration Utility**. For detailed information on this utility, refer to device driver guide for you module.

### KIDAQ Device Browser

Device Browser (*KPXIconf.exe*) displays the currently installed and detected KIDAQ hardware. The default installation directory for this utility is **C:\Keithley\KIDAQUtil**. It can also be found in the start menu under **Programs -> Keithley -> KIDAQ LabVIEW Driver -> Device Explorer**. The *KIDAQ device browser* main window is shown in [Figure C-3](#):



Figure C-3  
**Keithley PXI Devices Explorer**



The *Device Browser* main window contains three columns, *Device Type*, *Location (Bus/Device/Function)* and *Device Number*.

Device Type: Type of KIDAQ board installed

Location (Bus/Device/Function): The location the device is plugged into

Device Number: Number of device at PXI bus (Starts from 1)

Using this utility, user can view all of the KIDAQ devices connected to your system and get the device number corresponding to the device plugged on a specified PXI slot.

## KIDAQ LabVIEW VIs Overview

This section briefly describes each VI in the KIDAQ LabVIEW driver. The setup program detects the system (Windows® XP/2000), and installs the correct platform drivers to the system. All applications developed with KIDAQ LabVIEW are compatible across Windows® XP/2000.

You can find the detailed description of each VI using any of the following ways:

- Select the **Show Help** command in the **Help** menu in LabVIEW. Then, when you put the mouse cursor on KIDAQ LabVIEW VI, LabVIEW will show the description of the VI.
- Refer to [Appendix D](#) of this document
- Contact Keithley Instruments via phone, email, or on the web at [www.keithley.com](http://www.keithley.com) for further information

KIDAQ LabVIEW VIs are grouped into the following LabVIEW palettes:

- **Analog Input VIs**
- **Analog Output VIs**
  - Advanced Analog Output VIs

- **Digital I/O VIs**
  - Advanced Digital I/O
- **Timer/Counter VIs**
  - Intermediate Timer/Counter VIs
  - Advanced Timer/Counter VIs
- **Calibration and Configuration VIs**
- **Error Handler VI**

## Analog Input VIs

**KI AI Acquire Waveform:** Acquires a specified number of samples at a specified sample rate from a single input channel and returns the acquired data.

**KI AI Acquire Waveforms:** Acquires data from the specified channels and samples the channels at the specified scan rate.

**KI AI Sample Channel:** Measures the signal attached to the specified channel and returns the measured data.

**KI AI Sample Channels:** Performs a single reading from each of the specified channels.

**KI AI Clear:** The KI AI Clear VI stops an acquisition associated with taskID in.

**KI AI Config:** Configures an analog input operation for a specified set of channels.

**KI AI Read:** Reads data from a buffered data acquisition.

**KI AI Single Scan:** Returns one scan of data directly from the board analog input channels for a non-buffered acquisition.

**KI AI Start:** Starts a buffered analog input operation.

## Analog Output VIs

**KI AO Generate Waveform:** Generates a timed and buffered waveform for the given output channel at the specified update rate.

**KI AO Generate Waveforms:** Generates timed and buffered waveforms for the given output channels at the specified update rate.

**KI AO Update Channel:** Writes a specified value to an analog output channel.

**KI AO Update Channels:** Writes values to each of the specified analog output channels.

**KI AO Clear:** The KI AO Clear VI stops an analog output generation associated with taskID.

**KI AO Config:** Configures a buffered analog output operation.

**KI AO Start:** Starts a buffered analog output operation.

**KI AO Wait:** waits until the waveform generation of the task completes before returning.

**KI AO Write:** writes data into the buffer for a buffered analog output operation.

### Advanced Analog Output VIs

**KI AO Trigger and Gate Config:** Configures the trigger conditions for analog output operations.

## Digital I/O VIs

**KI Read from Digital Line:** Reads the logical state of a digital line on a digital channel that you configure.

**KI Read from Digital Port:** Reads a digital channel that you configure.

**KI Write to Digital Line:** Sets the output logic state of a digital line to high or low on a digital channel that you specify.

**KI Write to Digital Port:** Outputs a decimal pattern to a digital channel that you specify.

**KI DIO Clear:** Stops an acquisition associated with taskID.

**KI DIO Config:** Creates the taskID, establishes the handshake parameters, and allocates a buffer to hold the scans.

**KI DIO Read:** Calls the VI to read data from the internal transfer buffer and returns the data read in pattern.

**KI DIO Start:** Starts a buffered digital I/O operation.

**KI DIO Write:** Writes digital output data to the internal transfer buffer.

### Advanced Digital I/O VIs

**KI DIO Port Config:** Configures a digital Channel and returns a taskID to be used with Port VIs.

## Timer/Counter VIs

**KI Count Events or Time:** Configures one or two counters to count external events.

**KI Generate Delayed Pulse:** Configures and starts a counter to generate a single pulse with the specified delay and pulse-width.

**KI Generate Pulse Train:** Configures the specified counter to generate a continuous pulse-train.

**KI Measure Pulse Width or Period:** Measures the pulse-width (length of time a signal is high or low) or period (length of time between adjacent rising or falling edges) of a TTL signal.

### Intermediate Timer/Counter VIs

**KI Continuous Pulse Generator Config:** Configures a counter to generate a continuous TTL pulse-train.

**KI Counter Divider Config:** Configures the specified counter to divide a signal.

**KI Counter Read:** Reads the counter or counters identified by task ID.

**KI Counter Start:** Starts the counters identified by task ID.

**KI Counter Stop:** Stops a count operation immediately or conditionally on an input error.

**KI Delayed Pulse Generator Config:** Configures a counter to generate a single pulse with the specified delay and pulse-width.

**KI Down Counter or Divider Config:** Configures the specified counter to count down or divide a signal.

**KI Event or Time Counter Config:** Configures one or two counters to count external events.

**KI UpDown Counter Config:** Configures one counter to count edges in the signal on the specified counter's SOURCE pin or the number of cycles of a specified internal timebase signal.

## Advanced Timer/Counter VIs

**KI ICTR Control:** This VI control counters on the KIDAQ devices that use 82C54 chip.

## Calibration and Configuration VIs

**KI KPXI-DAQ Series Calibrate and Digitizer Series calibrate:** calibrates Keithley PXI DAQ device.

**KI Route Signal:** routes an internal signal to the specified I/O connector or SSI bus line, or to enable clock sharing through the SSI bus clock line.

**KI SSI Control:** Connects or disconnects trigger and timing signals between DAQ devices along the Real-Time System Integration (SSI) bus.

## Error Handler VI

**KI Error Handler:** explains non-zero error codes and shows dialog box with information about error.

## Distribution of Applications

To install an application using KIDAQ LabVIEW on another computer, you also must install the necessary driver files and supporting libraries on the target machine. You can create an automatic installer to install your program and all of the files needed to run that program or you can manually install the program and program files. Whichever installation method you choose, you must install the following files:

**NOTE** *Do not replace any files on the target computer if the file on the target computer has a newer version than the file you are installing.*

## Windows XP/2000

### LLB files

kidaq\_pci.llb in **C:\Keithley\KI-DAQ\LLB**

### Required support DLLs

Pci-iv.dll in **C:\Windows\system32**. This file should be copied to the same system32 directory on the target machine. On Windows 2000 the Windows directory is named winnt instead of Windows.

### Driver files

The corresponding driver files in **C:\Windows\system32\drivers**, e.g. **ksdaq4M2.sys** for **KPXI-SDAQ-4-2M**. These files should be copied to:

- **Windows\system32\drivers** directory (for Windows XP).
- **Winnt\system32\drivers** directory (for Windows 2000).

The corresponding INF file in **\Windows\inf**, e.g. **ksdaq4M2.inf** for **KPXI-SDAQ-4-2M**. These files should be copied to:

- **Windows\inf** directory (for Windows XP).
- **Winnt\inf** directory (for Windows 2000).

The location of the device configuration utility is: **C:\Keithley\KI-DAQ\Util\configdrv**

---

**KIDAQ<sup>®</sup>-LabVIEW Compatible Function Reference****In this appendix:**

<b>Topic</b>	<b>Page</b>
Introduction .....	D-2
Hardware support .....	D-2
KPXI-DIO series: .....	D-2
KPXI-DAQ series: .....	D-2
Digitizer series: .....	D-2
Analog input VIs .....	D-3
Easy analog input VIs .....	D-3
Intermediate analog input VIs .....	D-7
Analog output VIs .....	D-21
Easy analog output VIs .....	D-21
Intermediate analog output VIs .....	D-24
Advanced analog output VIs .....	D-32
Digital I/O VIs .....	D-33
Easy Digital I/O VIs .....	D-33
Intermediate Digital I/O VIs .....	D-37
Advanced Digital I/O VIs .....	D-45
Counter VIs .....	D-46
Easy Counter VIs .....	D-46
Intermediate Counter VIs .....	D-50
Advanced Counter VIs .....	D-63
Calibration and Configuration VIs .....	D-67
Calibration VIs .....	D-67
Other Calibration and Configuration VIs .....	D-68
Service VIs .....	D-70
Error Codes .....	D-71
AI Range Codes .....	D-73
AI Data Format .....	D-76

## Introduction

This function reference provides a detailed description of LabVIEW<sup>1</sup> compatible interfaces for Keithley Instruments PXI DAQ modules.

## Hardware support

Keithley Instruments will periodically upgrade KIDAQ LabVIEW to add support for new Keithley Instruments PXI data acquisition modules. This release of KIDAQ LabVIEW supports the following hardware:

### KPXI-DIO series:

- **KPXI-DIO-16-16:** 32 channels isolated Digital I/O card
- **KPXI-DIO-48:** 48-bit digital I/O card
- **KPXI-RDI-8-16:** 8 relay output and 16 isolated input card
- **KPXI-DIO-32-80M:** 40 Mbytes/sec Ultra-high speed 32 channels digital I/O module with bus mastering DMA transfer supporting scatter gather technology
- **KPXI-DIO-32-32:** 32 isolated channels DI & 32 isolated channels DO card
- **KPXI-DIO-64-0:** 64 isolated channels DI card
- **KPXI-DIO-0-64:** 64 isolated channels DO card

### KPXI-DAQ series:

- **KPXI-SDAQ-4-2M:** 2MHz 4 channels simultaneous A/D and 2 channels D/A output device with bus mastering DMA transfer capability
- **KPXI-SDAQ-4-500K:** 500kHz 4 channels simultaneous A/D and 2 channels D/A output device with bus mastering DMA transfer capability
- **KPXI-DAQ-64-3M:** 3MHz 64 channels multiplexed A/D and 2 channels D/A output device with bus mastering DMA transfer capability
- **KPXI-DAQ-64-500K:** 500kHz 64 channels multiplexed A/D and 2 channels D/A output device with bus mastering DMA transfer capability
- **KPXI-DAQ-64-250K:** 250kHz 64 channels multiplexed A/D and 2 channels D/A output device with bus mastering DMA transfer capability
- **KPXI-DAQ-96-3M:** 3MHz 96 channels multiplexed A/D device with bus mastering DMA transfer capability
- **KPXI-AO-4-1M:** High Performance 4 channels analog output Multi-function device with bus mastering DMA transfer capability
- **KPXI-AO-8-1M:** High Performance 8 channels analog output Multi-function device with bus mastering DMA transfer capability

### Digitizer series:

- **KPXI-AI-2-65M:** 130MHz or 2 channels simultaneous A/D digitizer with bus mastering DMA transfer capability

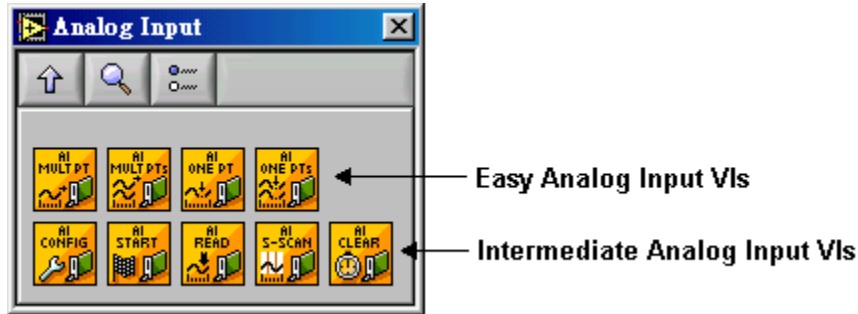
---

1. LabVIEW™ is a trademark of the National Instruments Corporation. All other trademarks are the property of their respective owners.

# Analog input VIs

Analog Input VIs (virtual instruments) are available in the Analog Input palette (Figure D-1).

Figure D-1  
Analog input palette



## Easy analog input VIs

### KI AI acquire waveform

This VI acquires a specified number of samples at a specified sample rate from a single input channel and returns the acquired data. This VI performs a timed measurement of a waveform on a single analog input channel. If an error occurs, a dialog box appears providing error information.

Table D-1  
KI AI acquire waveform

<b>I16</b>	<b>device:</b> Number of the device (beginning from 1). The utility <i>Device Browser</i> can be used to get the information of current device configuration.
<b>U16</b>	<b>sub type:</b> is the sub-type of the device you assigned to the Keithley Instruments PXI device during configuration.
<b>abc</b>	<b>channel:</b> identifies the analog input channel you want to measure. The default input is channel 0. The valid channel for each Keithley Instruments PXI device is as follows: <ul style="list-style-type: none"> <li>• <b>KPXI-SDAQ-4-2M:</b> 0 through 3</li> <li>• <b>KPXI-SDAQ-4-500K:</b> 0 through 3</li> <li>• <b>KPXI-DAQ-64-3M:</b> 0 through 63</li> <li>• <b>KPXI-DAQ-64-500K:</b> 0 through 63</li> <li>• <b>KPXI-DAQ-64-250K:</b> 0 through 63</li> <li>• <b>KPXI-DAQ-96-3M:</b> 0 through 95</li> <li>• <b>KPXI-AO-4-1M:</b> 0 through 7</li> <li>• <b>KPXI-AO-8-1M:</b> 0 through 3</li> <li>• <b>KPXI-AI-2-65M:</b> 0 through 1</li> </ul>
<b>I32</b>	<b>number of samples:</b> is the number of samples the VI acquires. The default for this parameter is 1000 samples, except KPXI-AI-2-65M. For KPXI-AI-2-65M, the default value is 1024.
<b>SGL</b>	<b>sample rate:</b> is the requested number of samples per second for the analog input. The default for this parameter is a rate of 1000 samples/s.

Table D-1 (continued)

**KI AI acquire waveform**

<b>SGL</b>	<b>high limit:</b> is the maximum scaled data in Volts. The default input is 0. If both high limit and low limit are 0, high limit keeps the default settings.
<b>SGL</b>	<b>low limit:</b> is the minimum scaled data in Volts. The default input is 0. If both high limit and low limit are 0, low limit keeps the default settings.
<b>U16</b>	<b>input config:</b> defines the mode that the channel should be scanned. 0: No change (default input) 1: Differential (default setting) 2: Referenced single-ended 3: Nonreferenced single-ended
<b>[ SGL ]</b>	<b>waveform:</b> contains scaled analog input data.
<b>SGL</b>	<b>actual sample period:</b> is the actual interval between samples, which is the inverse of the actual sample rate. The actual sample period can differ from the requested sample rate, depending on the capabilities of the hardware.

**KI AI acquire waveforms**

Acquires data from the specified channels at the specified scan rate. This VI performs a timed measurement of multiple waveforms on the specified analog input channels. If an error occurs, a dialog box appears, giving you the error information.

Table D-2

**KI AI acquire waveforms**

<b>I16</b>	<b>device:</b> Number of the device (beginning from 1). The utility <b>Device Browser</b> can be used to get the information of current device configuration.
<b>U16</b>	<b>sub type:</b> is the sub-type of the device you assigned to the Keithley Instruments PXI device during configuration. Only the following series of devices need to specify the sub type.



Table D-2 (continued)  
**KI AI acquire waveforms**

<p>[abc]</p>	<p><b>channels:</b> specifies the set of analog input channels you want to measure. The order of the channels in the scan list defines the order in which the channels are scanned. If x, y, and z refer to channels, you can specify a list of channels in a single element by “x,y,z”. If x refers to the first channel in a consecutive channel range and y refers to the last channel, you can specify the range by “x:y”. The default input is channel 0.                  The valid channel order for acquiring data is as follows:  <b>KPXI-SDAQ-4-2M:</b> numbers in channels must be within 0 and 3 and the continuous scan sequence is ascending with consecutive channels.  <b>KPXI-SDAQ-4-500K:</b> numbers in channels must be within 0 and 3 and the continuous scan sequence is ascending with consecutive channels.  <b>KPXI-DAQ-64-3M:</b> numbers in channels must be within 0 and 63 and there is no restriction of channel order setting; therefore you can set the channel order as you wish.  <b>KPXI-DAQ-64-500K:</b> numbers in channels must be within 0 and 63 and there is no restriction of channel order setting; therefore you can set the channel order as you wish.  <b>KPXI-DAQ-64-250K:</b> numbers in channels must be within 0 and 63 and there is no restriction of channel order setting; therefore you can set the channel order as you wish.  <b>KPXI-DAQ-96-3M:</b> numbers in channels must be within 0 and 95 and there is no restriction of channel order setting; therefore you can set the channel order as you wish.  <b>KPXI-AO-4-1M:</b> numbers in channels must be within 0 and 3 and the continuous scan sequence is ascending with consecutive channels.  <b>KPXI-AO-8-1M:</b> numbers in channels must be within 0 and 7 and the continuous scan sequence is ascending with consecutive channels.  <b>KPXI-AI-2-65M:</b> numbers in channels must be within 0 and 1 and the continuous scan sequence is ascending with consecutive channels.</p>
<p>[I32]</p>	<p><b>number of samples/ch:</b> is the number of samples per channel. The default is 1000 samples/ch, except KPXI-AI-2-65M. For KPXI-AI-2-65M, the default value is 1024 samples/ch.</p>
<p>[SGL]</p>	<p><b>scan rate:</b> is the requested number of scans per second. The default is 1000 scans/s. A scan is one sample/channel.</p>
<p>[SGL]</p>	<p><b>high limit:</b> is the maximum scaled data in Volts. The default input is 0. If both high limit and low limit are 0, high limit keeps the default settings.</p>
<p>[SGL]</p>	<p><b>low limit:</b> is the minimum scaled data in Volts. The default input is 0. If both high limit and low limit are 0, low limit keeps the default settings.</p>
<p>[U16]</p>	<p><b>input config:</b> defines the mode that the channel should be scanned.                  0: No change (default input)                  1: Differential (default setting)                  2: Referenced single-ended                  3: Nonreferenced single-ended</p>
<p>[SGL]</p>	<p><b>waveforms:</b> is a 2D array that contains analog input data in Volts.</p>
<p>[SGL]</p>	<p><b>actual scan period:</b> is the actual interval between scans, which is the inverse of the actual scan rate. The actual scan period can differ from the requested scan rate, depending on the capabilities of the hardware.</p>

### KI AI sample channel

This VI performs a single, un-timed measurement of a channel. It measures the signal attached to the specified channel and returns the measured data (in Volts). If an error occurs, a dialog box appears, giving you the error information.

Table D-3

#### KI AI sample channel

<b>I16</b>	<b>device:</b> Number of the device (beginning from 1). The utility <b>Device Browser</b> can be used to get the information of current device configuration.
<b>U16</b>	<b>sub type:</b> is the sub-type of the device you assigned to the Keithley Instruments PXI device during configuration.
<b>abc</b>	<b>channel:</b> identifies the analog input channel you want to measure. The default input is channel 0. The valid channel for each Keithley Instruments PXI device is as follows: <b>KPXI-SDAQ-4-2M:</b> 0 through 3 <b>KPXI-SDAQ-4-500K:</b> 0 through 3 <b>KPXI-DAQ-64-3M:</b> 0 through 63 <b>KPXI-DAQ-64-500K:</b> 0 through 63 <b>KPXI-DAQ-64-250K:</b> 0 through 63 <b>KPXI-DAQ-96-3M:</b> 0 through 95 <b>KPXI-AO-4-1M:</b> 0 through 3 <b>KPXI-AO-8-1M:</b> 0 through 7
<b>SGL</b>	<b>high limit:</b> is the maximum scaled data in Volts. The default input is 0. If both high limit and low limit are 0, high limit keeps the default setting.
<b>SGL</b>	<b>low limit:</b> is the minimum scaled data in Volts. The default input is 0. If both high limit and low limit are 0, low limit keeps the default setting.
<b>U16</b>	<b>input config:</b> defines the mode that the channel should be scanned. 0: No change (default input) 1: Differential (default setting) 2: Referenced single-ended 3: Nonreferenced single-ended
<b>SGL</b>	<b>sample:</b> contains the scaled analog input data for the specified channel.

### KI AI sample channels






This VI measures a single value from each of the specified analog input channels. If an error occurs, a dialog box appears, giving you the error information.

Table D-4

#### KI AI sample channels

<b>I16</b>	<b>device:</b> Number of the device (beginning from 1). The utility <b>Device Browser</b> can be used to get the information of current device configuration.
<b>U16</b>	<b>sub type:</b> is the sub-type of the device you assigned to the Keithley Instruments PXI device during configuration.

Table D-4 (continued)  
**KI AI sample channels**

	<p><b>channels:</b> specifies the set of analog input channels you want to measure. The order of the channels in the scan list defines the order in which the channels are scanned. If x, y, and z refer to channels, you can specify a list of channels in a single element by “x,y,z”. If x refers to the first channel in a consecutive channel range and y refers to the last channel, you can specify the range by “x:y”. The default input is channel 0.</p> <p>The valid channel order for acquiring data is as follows:</p> <p><b>KPXI-SDAQ-4-2M:</b> numbers in channels must be within 0 and 3 and the continuous scan sequence is ascending with consecutive channels.</p> <p><b>KPXI-SDAQ-4-500K:</b> numbers in channels must be within 0 and 3 and the continuous scan sequence is ascending with consecutive channels.</p> <p><b>KPXI-DAQ-64-3M:</b> numbers in channels must be within 0 and 63 and there is no restriction of channel order setting; therefore you can set the channel order as you wish.</p> <p><b>KPXI-DAQ-64-500K:</b> numbers in channels must be within 0 and 63 and there is no restriction of channel order setting; therefore you can set the channel order as you wish.</p> <p><b>KPXI-DAQ-64-250K:</b> numbers in channels must be within 0 and 63 and there is no restriction of channel order setting; therefore you can set the channel order as you wish.</p> <p><b>KPXI-DAQ-96-3M:</b> numbers in channels must be within 0 and 95 and there is no restriction of channel order setting; therefore you can set the channel order as you wish.</p> <p><b>KPXI-AO-4-1M:</b> numbers in channels must be within 0 and 3 and the continuous scan sequence is ascending with consecutive channels.</p> <p><b>KPXI-AO-8-1M:</b> numbers in channels must be within 0 and 7 and the continuous scan sequence is ascending with consecutive channels.</p>
	<p><b>high limit:</b> is the maximum scaled data in Volts. The default input is 0. If both high limit and low limit are 0, high limit keeps the default setting.</p>
	<p><b>low limit:</b> is the minimum scaled data in Volts. The default input is 0. If both high limit and low limit are 0, low limit keeps the default setting.</p>
	<p><b>input config:</b> defines the mode that the channel should be scanned.</p> <ul style="list-style-type: none"> <li>0: No change (default input)</li> <li>1: Differential (default setting)</li> <li>2: Referenced single-ended</li> <li>3. Nonreferenced single-ended</li> </ul>
	<p><b>sample:</b> is a 1D array that contains scaled analog input data.</p>

### Intermediate analog input VIs

#### KI AI clear

This VI stops an acquisition operation. Before beginning a new acquisition, you must call the KI AI Config VI.

Table D-5  
**KI AI clear**











	<p><b>taskID in:</b> identifies the group and the I/O operation.</p>
	<p><b>error in (no error):</b> describes error conditions occurring before the VI executes. If an error has already occurred, the VI returns the value of the error in cluster in error out.</p>

Table D-5 (continued)

**KI AI clear**

		<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
		<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
		<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.
		<b>taskID out:</b> has the same value as taskID in.
		<b>error out:</b> contains error information. If the error in cluster indicated an error, the error out cluster contains the same information. Otherwise, error out describes the error status of this VI.
		<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
		<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
		<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.

**KI AI config**

Configures a buffered analog input operation, including configuring the hardware and allocating a buffer.

Table D-6

**KI AI config**













	<p><b>interchannel delay:</b> For devices with both scan and channel clocks (KPXI-DAQ series devices), you can use interchannel delay to specify the waiting time between sampling channels within a scan. Select a default interchannel delay automatically, giving the hardware time to settle between channels. The default value for interchannel delay is -1.0, which tells the AI Config VI to use the channel clock rate LabVIEW selects.</p>	
	<p><b>sub type:</b> is the sub-type of the device you assigned to the Keithley Instruments PXI device during configuration.</p>	
	<p><b>measurement mode structure:</b> This input is not used by Keithley Instruments PXI devices and is ignored.</p>	
		<p><b>measurement mode:</b> is not used</p>
		<p><b>reserved:</b> is not used</p>
	<p><b>coupling &amp; input config:</b> is an array of clusters. Each array element contains the configuration for the channel or channels specified by the corresponding element of the channels array. KIDAQ LabVIEW uses only input config. The default for the coupling &amp; input config array is an empty array, which means the parameters keep their default settings.</p>	
		<p><b>coupling:</b> This input is not used by Keithley Instruments PXI devices and is ignored.</p>
		<p><b>input config:</b> defines the mode that the channel should be scanned.                  0: No change (default input)                  1: Differential (default setting)                  2: Referenced single-ended                  3: Nonreferenced single-ended</p>
	<p><b>input limits:</b> is an array of clusters. Each array element contains the expected signal limits for the channels specified by the corresponding element of channels. If there are fewer elements in this array than in channels, the VI uses the last array element for the rest of the channels. The default for the input limits array is an empty array, which means the input limits keep their default settings.</p>	
		<p><b>high limit:</b> is the maximum scaled data in Volts. The default input is 0.</p>
		<p><b>low limit:</b> is the minimum scaled data in Volts. The default input is 0.</p>
	<p><b>device:</b> Number of the device at PXI-bus (beginning from 1). The utility <b>Device Browser</b> can be used to get the information of current device configuration.</p>	

Table D-6 (continued)

## KI AI config

















	<p><b>channels:</b> specifies the set of analog input channels. The order of the channels in the scan list defines the order in which the channels are scanned. channels is an array of strings. You can use one channel entry per element or specify the entire scan list in a single element, or use any combination of these two methods. If x, y, and z refer to channels, you can specify a list of channels in a single element by "x,y,z". If x refers to the first channel in a consecutive channel range and y refers to the last channel, you can specify the range by "x:y".</p> <p>The valid channel order for acquiring data is as follows:</p> <p><b>KPXI-SDAQ-4-2M:</b> numbers in channels must be within 0 and 3 and the continuous scan sequence is ascending with consecutive channels.</p> <p><b>KPXI-SDAQ-4-500K:</b> numbers in channels must be within 0 and 3 and the continuous scan sequence is ascending with consecutive channels.</p> <p><b>KPXI-DAQ-64-3M:</b> numbers in channels must be within 0 and 63 and there is no restriction of channel order setting; therefore you can set the channel order as you wish.</p> <p><b>KPXI-DAQ-64-500K:</b> numbers in channels must be within 0 and 63 and there is no restriction of channel order setting; therefore you can set the channel order as you wish.</p> <p><b>KPXI-DAQ-64-250K:</b> numbers in channels must be within 0 and 63 and there is no restriction of channel order setting; therefore you can set the channel order as you wish.</p> <p><b>KPXI-DAQ-96-3M:</b> numbers in channels must be within 0 and 95 and there is no restriction of channel order setting; therefore you can set the channel order as you wish.</p> <p><b>KPXI-AO-4-1M:</b> numbers in channels must be within 0 and 7 and the continuous scan sequence is ascending with consecutive channels.</p> <p><b>KPXI-AO-8-1M:</b> numbers in channels must be within 0 and 3 and the continuous scan sequence is ascending with consecutive channels.</p> <p><b>KPXI-AI-2-65M:</b> numbers in channels must be within 0 and 1 and the continuous scan sequence is ascending with consecutive channels.</p>
	<p><b>buffer size:</b> is the number of scans you want the buffer to hold. The default for this parameter is 1000 scans, except KPXI-AI-2-65M. For KPXI-AI-2-65M, the default value is 1024 scans.</p>
	<p><b>group:</b> is the number, from 0 to 15, that you assign to the specified set of channels. The default input and setting for group is 0. If you only have one acquisition for this device, leave this input unwired and use group 0.</p>
	<p><b>error in (no error):</b> describes error conditions occurring before the VI executes. If an error has already occurred, the VI returns the value of the error in cluster in error out.</p>
	<p><b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.</p>
	<p><b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.</p>

Table D-6 (continued)  
**KI AI config**

		<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.
		<b>number of buffers:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
		<b>allocation mode:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
		<b>number of AMUX boards:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
		<b>taskID:</b> identifies the group and the I/O operation.
		<b>number of channels:</b> is the total number of channels in the group.
		<b>error out:</b> contains error information. If the error in cluster indicated an error, the error out cluster contains the same information. Otherwise, error out describes the error status of this VI.
		<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
		<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
		<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.

## KI AI Read

This VI reads specified number of scans of data from a buffered analog input acquisition.

KI AI Read is a polymorphic VI that you can configure to output the following kinds of data:

- 2-byte Binary Array (KPXI-AI-2-65M and KDAQ-DRVR series devices)([Table D-7](#))
- Scaled and 2-byte Binary Arrays (KPXI-AI-2-65M and KDAQ-DRVR series devices)([Table D-8](#))
- Scaled Array ([Table D-9](#))

Table D-7

### 2-byte binary array














	<b>conditional retrieval specification:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
	<b>mode:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
	<b>channel index:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
	<b>slope:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
	<b>level:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
	<b>hysteresis:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
	<b>skip count:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
	<b>offset:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
	<b>taskID in:</b> identifies the group and the I/O operation.
	<b>number of scans to read:</b> is the number of scans the VI is to retrieve from the acquisition buffer. The default input is -1, which set number of scans to read equal to the value of the number of scans to acquire parameter when the KI AI Start was called. If number of scans to read is -1 and number of scans to acquire is 0, KIDAQ LabVIEW sets number of scans to read to be the half of the buffer size.
	<b>time limit in sec:</b> is the time limit for the read operation. The default input is -1.0, which means KIDAQ LabVIEW calculates a time limit based on the value of number of scans to read and the scan rate. If the scan rate is unknown, the VI uses 1 second as the time limit. The resolution of the timeout clock is about 55 ms.
	<b>error in (no error):</b> describes error conditions occurring before the VI executes. If an error has already occurred, the VI returns the value of the error in cluster in error out.
	<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.



Table D-7 (continued)  
**2-byte binary array**
















		<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
		<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.
	<b>read/search position:</b> This input is not used by Keithley Instruments PXI devices and is ignored. The starting point for the read is the position where the read mark points to. Initially, the read mark points to the beginning of the acquisition buffer. As you retrieve data from the buffer using this VI, KIDAQ LabVIEW increments the read mark to point to the next block of data to be read.	
		<b>position:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
		<b>read offset:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
	<b>scan backlog:</b> is the amount of data remaining in the buffer after this VI completes.	
	<b>number read:</b> is the number of scans returned. This number is identical to number of scans to read unless an error or timeout appears or the VI reaches the end of the data.	
	<b>taskID out:</b> has the same value as <i>taskID in</i> .	
 or 	<b>binary data:</b> is a 2D array that contains unscaled analog input data.	
	<b>retrieval complete:</b> is TRUE when the acquisition finishes and no backlog data remains.	
	<b>error out:</b> contains error information. If the error in cluster indicated an error, the error out cluster contains the same information. Otherwise, error out describes the error status of this VI.	
	<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.	
		<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
		<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.

Table D-8  
Scaled and Binary Arrays
















	<b>conditional retrieval specification:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
	<b>mode:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
	<b>channel index:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
	<b>slope:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
	<b>level:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
	<b>hysteresis:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
	<b>skip count:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
	<b>offset:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
	<b>taskID in:</b> Identifies the group and the I/O operation.
	<b>number of scans to read:</b> is the number of scans the VI is to retrieve from the acquisition buffer. The default input is -1, which set number of scans to read equal to the value of the number of scans to acquire parameter when the KI AI Start was called. If number of scans to read is -1 and number of scans to acquire is 0, KIDAQ LabVIEW sets number of scans to read to be the half of the buffer size.
	<b>time limit in sec:</b> is the time limit for the read operation. The default input is -1.0, which means KIDAQ LabVIEW calculates a time limit based on the value of number of scans to read and the scan rate. If the scan rate is unknown, the VI uses 1 second as the time limit. The resolution of the timeout clock is about 55 ms.
	<b>error in (no error):</b> describes error conditions occurring before the VI executes. If an error has already occurred, the VI returns the value of the error in cluster in error out.
	<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
	<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
	<b>source:</b> Identifies where an error occurred. The source string is usually the name of the VI that produced the error.

Table D-8 (continued)  
**Scaled and Binary Arrays**














	<p><b>read/search position:</b> This input is not used by Keithley Instruments PXI devices and is ignored.                  The starting point for the read is the position where the read mark point to. Initially, the read mark points to the beginning of the acquisition buffer. As you retrieve data from the buffer using this VI, KIDAQ LabVIEW increments the read mark to point to the next block of data to be read.</p>
	<p><b>position:</b> This input is not used by Keithley Instruments PXI devices and is ignored.</p>
	<p><b>read offset:</b> This input is not used by Keithley Instruments PXI devices and is ignored.</p>
	<p><b>scan backlog:</b> is the amount of data remaining in the buffer after this VI completes.</p>
	<p><b>number read:</b> is the number of scans returned. This number is identical to number of scans to read unless an error or timeout appears or the VI reaches the end of the data.</p>
	<p><b>taskID out:</b> has the same value as <i>taskID in</i>.</p>
<p data-bbox="391 930 483 976">[ I16 ]</p> <p data-bbox="391 982 412 1008">or</p> 	<p><b>binary data:</b> is a 2D array that contains unscaled analog input data.</p>
	<p><b>scaled data:</b> is a 2D array that contains analog input data in Volts.</p>
	<p><b>retrieval complete:</b> is TRUE when the acquisition finishes and no backlog data remains.</p>
	<p><b>error out:</b> contains error information. If the error in cluster indicated an error, the error out cluster contains the same information. Otherwise, error out describes the error status of this VI.</p>
	<p><b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.</p>
	<p><b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.</p>
	<p><b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.</p>

Table D-9  
Scaled Array




























	<b>conditional retrieval specification:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
	<b>mode:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
	<b>channel index:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
	<b>slope:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
	<b>level:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
	<b>hysteresis:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
	<b>skip count:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
	<b>offset:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
	<b>taskID in:</b> Identifies the group and the I/O operation.
	<b>number of scans to read:</b> is the number of scans the VI is to retrieve from the acquisition buffer. The default input is -1, which set number of scans to read equal to the value of the number of scans to acquire parameter when the KI AI Start was called. If number of scans to read is -1 and number of scans to acquire is 0, KIDAQ LabVIEW sets number of scans to read to be the half of the buffer size.
	<b>time limit in sec:</b> is the time limit for the read operation. The default input is -1.0, which means KIDAQ LabVIEW calculates a time limit based on the value of number of scans to read and the scan rate. If the scan rate is unknown, the VI uses 1 second as the time limit. The resolution of the timeout clock is about 55 ms.
	<b>error in (no error):</b> describes error conditions occurring before the VI executes. If an error has already occurred, the VI returns the value of the error in cluster in error out.
	<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
	<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
	<b>source:</b> Identifies where an error occurred. The source string is usually the name of the VI that produced the error.

Table D-9 (continued)

**Scaled Array**

	<b>read/search position:</b> This input is not used by Keithley Instruments PXI devices and is ignored. The starting point for the read is the position where the read mark point to. Initially, the read mark points to the beginning of the acquisition buffer. As you retrieve data from the buffer using this VI, KIDAQ LabVIEW increments the read mark to point to the next block of data to be read.
	<b>position:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
	<b>read offset:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
	<b>scan backlog:</b> is the amount of data remaining in the buffer after this VI completes.
	<b>number read:</b> is the number of scans returned. This number is identical to number of scans to read unless an error or timeout appears or the VI reaches the end of the data.
	<b>taskID out:</b> has the same value as taskID in.
	<b>scaled data:</b> is a 2D array that contains analog input data in Volts.
	<b>retrieval complete:</b> is TRUE when the acquisition finishes and no backlog data remains.
	<b>error out:</b> contains error information. If the error in cluster indicated an error, the error out cluster contains the same information. Otherwise, error out describes the error status of this VI.
	<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
	<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
	<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.

**KI AI Single Scan**

This VI returns one scan of data from the analog input channels for a non-buffered acquisition.

Table D-10

**KI AI single scan**

















Binary Array	
	<b>taskID in:</b> identifies the group and the I/O operation.

Table D-10 (continued)

## KI AI single scan

	<b>opcode:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
	<b>time limit in sec:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
	<b>error in (no error):</b> describes error conditions occurring before the VI executes. If an error has already occurred, the VI returns the value of the error in cluster in error out.
	<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
	<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
	<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.
	<b>data remaining:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
	<b>taskID out:</b> has the same value as taskID in.
 or 	<b>binary data:</b> contains the unscaled binary data in Volts. The array index represents the channel.
	<b>acquisition state:</b> This input is not used by Keithley Instruments PXI devices and is ignored
	<b>error out:</b> contains error information. If the error in cluster indicated an error, the error out cluster contains the same information. Otherwise, error out describes the error status of this VI.
	<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
	<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
	<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.

**KI AI start**

Starts a buffered analog input operation. This VI sets the scan rate, the number of scans to acquire, the conversion clock source, and the trigger conditions. The VI then starts an acquisition.

Table D-11  
**KI AI start**












	<b>taskID in:</b> identifies the group and the I/O operation.
	<b>number of scans to acquire:</b> is the total number of scans to acquire. A scan is one point per channel. With the default input -1, the device acquires exactly one buffer of data. The buffer size input to the KI AI Config VI determines the size of the buffer. The number of total scans includes any pretrigger scans requested. If you set number of scans to acquire to 0, the device acquires data indefinitely into the buffer until you stop the acquisition with the KI AI Clear VI. In this case, the VI ignores the pretrigger scans input. For KPXI-AI-2-65M, the number of scans to acquire has to be equal to the buffer size input to the KI AI Config VI.
	<b>scan rate:</b> is the number of scans/s to acquire. This is equivalent to the sampling rate per channel. The default for this parameter is 1000 scans/s. If you enter 0, the on-board internal clock is disabled and the external clock is used.
	<b>error in (no error):</b> describes error conditions occurring before the VI executes. If an error has already occurred, the VI returns the value of the error in cluster in error out.
	<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
	<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
	<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.
	<b>edge or slope:</b> 0: Do not change the default setting (default input). 1: Leading edge for digital trigger; positive slope for analog trigger. 2: Trailing edge for digital trigger; negative slope for analog trigger.
	<b>pretrigger scans:</b> is the number of scans you want to save in the buffer before the trigger. The default for this parameter is 0, which means no data before the trigger is saved.
	<b>trigger type:</b> specifies the type of trigger to start or stop the acquisition. 0: No trigger (default input). 1: Analog trigger (default setting). 2: Digital trigger. 3: SSI digital start trigger (for KPXI-AI-2-65M, the signal is through PXI trigger bus 3) and analog trigger (for the applications need both start and stop triggers, e.g. middle trigger or pre-trigger mode of operation). 4: SSI digital start trigger (for KPXI-AI-2-65M, the signal is through PXI trigger bus 3) and digital trigger (for the applications need both start and stop triggers, e.g. middle trigger or pre-trigger mode of operation).
	<b>number of buffers to acquire:</b> This input is not used by Keithley Instruments PXI devices and is ignored. There is always only one buffer.

Table D-11 (continued)

## KI AI start





















	<p><b>scan clock source:</b> identifies the A/D clock source.</p> <p>0: Do not change the clock source setting (default input).</p> <p>1: An internal timebase is used (default setting).</p> <p>2: You supply a signal through the I/O connector 1 (for KPXI-DAQ series devices, the signal is from AF10/AF11 and for KPXI-AI-2-65M the signal is from CLK IN connector).</p> <p>3: You supply a signal through the I/O connector 2 (for KPXI-DAQ series devices and KPXI-AI-2-65M, the signal is from SSI cable).</p> <p>4: An internal timebase with double edged enabled (only available for KPXI-AI-2-65M).</p> <p>5: You supply a signal through the I/O connector 1 with double edged enabled (only available for KPXI-AI-2-65M).</p> <p>6: a signal from SSI cable with double edged enabled (only available for KPXI-AI-2-65M).</p> <p>7: external timebase from SSI cable (for KPXI-DAQ series devices, the timebase is 40MHz and for KPXI-AI-2-65M, the time base is 60 MHz).</p> <p>8: both conversion signal and external timebase from SSI cable (only available for KPXI-DAQ series devices. The timebase is 40MHz).</p>	
	<p><b>analog chan and level:</b> contains the following parameter.</p>	
		<p><b>trigger channel:</b> specifies where the trigger comes from.</p> <p>When trigger type is 1 (analog trigger), the default for trigger channel is 0, i.e. analog input channel 0.</p> <p>When trigger type is 2 (digital trigger):</p> <p>0: external digital pin (default).</p> <p>1 : the signal from SSI cable.</p> <p>2: both start and stop trigger signals are from SSI cable (available for KPXI-DAQ series devices or KPXI-AI-2-65M).</p> <p>3~10: the signal is from PXI trigger bus 0 to 7. (only available for KPXI-AI-2-65M)</p> <p>11: the signal is PXI_START signal. (only available for KPXI-AI-2-65M)</p>
		<p><b>level:</b> level (measured in Volts) which analog source must cross for a trigger to occur. The default input for level is 0.0.</p>
	<p><b>additional trig params:</b> cluster contains the following parameters:</p>	
		<p><b>hysteresis:</b> This input is not used by Keithley Instruments PXI devices and is ignored.</p>
		<p><b>coupling:</b> This input is not used by Keithley Instruments PXI devices and is ignored.</p>
		<p><b>delay:</b> specifies how long the device waits after a trigger occurs before sampling data. You express delay in seconds. The default input and setting are 0.0s (no delay).</p>
		<p><b>skip count:</b> This input is not used by Keithley Instruments PXI devices and is ignored.</p>
		<p><b>time limit:</b> This input is not used by Keithley Instruments PXI devices and is ignored.</p>
	<p><b>taskID out:</b> has the same value as <i>taskID in</i>.</p>	



Table D-11 (continued)  
**KI AI start**

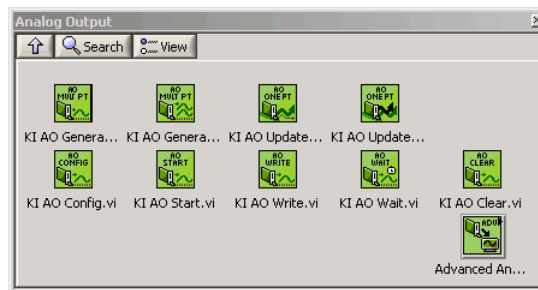
	<b>error out:</b> contains error information. If the error in cluster indicated an error, the error out cluster contains the same information. Otherwise, error out describes the error status of this VI.	
	<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.	
	<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.	
	<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.	
	<b>actual scan rate:</b> may differ slightly from the requested scan rate, depending on the hardware capabilities.	
	<b>actual trigger params:</b> cluster may differ slightly from the requested trigger inputs, depending on the hardware capabilities. It contains the following parameters.	
	<b>actual level:</b> is the analog trigger level the VI used.	
	<b>actual hysteresis:</b> This input is not used by Keithley Instruments PXI devices and is ignored.	
	<b>actual delay:</b> is the delay the VI used.	

## Analog output VIs

### Easy analog output VIs

Analog Output VIs (virtual instruments) are available in the Analog Output palette ([Figure D-2](#)).

Figure D-2  
**Analog output palette**











### KI AO generate waveform

Generates a timed and buffered waveform for the given output channel at the specified update rate. The KI AO Generate Waveform VI generates a waveform on a specified analog output channel. It does not return until the generation is complete.

**NOTE** This VI is not supported for Keithley KDIO Series devices.

Table D-12

**KI AO generate waveform**

	<b>device:</b> Number of the device (beginning from 1). The utility <b>Device Browser</b> can be used to get the information of current device configuration.
	<b>sub type:</b> is the sub-type of the device you assigned to the Keithley Instruments PXI device during configuration.
	<b>channel:</b> identifies the analog output channel. The default input is channel 0. The valid channel for each Keithley Instruments PXI series device is as follows: <b>KPXI-SDAQ-4-500K/KPXI-SDAQ-4-2M/KPXI-DAQ-64-3M/KPXI-DAQ-64-500K/</b> <b>KPXI-DAQ-64-250K/KPXI-DAQ-96-3M:</b> 0 or 1 <b>KPXI-AO-4-1M:</b> 0 through 3 <b>KPXI-AO-8-1M:</b> 0 through 7
	<b>high limit:</b> is the highest expected level of the signal in Volts you want to generate.
	<b>low limit:</b> is the lowest expected level of the signal in Volts you want to generate.
	<b>reference source:</b> is the internal/external setting of the reference voltage for this channel. 0: Do not change the reference source setting (default input). 1: Internal (default setting). 2: External.
	<b>update rate:</b> is the number of updates to generate per second. The default rate is 1000 update/s.
	<b>waveform:</b> is a 1D array that contains analog output data to be written the specified channel expressed in Volts. The data must be supplied.

**KI AO generate waveforms**

Generates timed and buffered waveforms for the given output channels at the specified update rate. The KI AO Generate Waveforms VI generates waveforms on specified analog output channels. It does not return until the generation is complete.

**NOTE** This VI is not supported for Keithley KDIO Series devices.

Table D-13

**KI AO generate waveforms**









	<b>device:</b> Number of the device (beginning from 1). The utility <b>Device Browser</b> can be used to get the information of current device configuration.
	<b>sub type:</b> is the sub-type of the device you assigned to the Keithley Instruments PXI device during configuration.

Table D-13 (continued)  
**KI AO generate waveforms**

	<b>channels:</b> Specifies the set of analog output channels you want to use. If x, y, and z refer to channels, you can specify a list of channels by separating the individual channels with commas (for example, x,y,z). If x refers to the first channel in a consecutive channel range and y refers to the last channel, you can specify the range by separating the first and last channels by a colon (for example, x:y). See KI AO generate waveform above for available channels on each module.
	<b>high limit:</b> is the highest expected level of the signal in Volts you want to generate.
	<b>low limit:</b> is the lowest expected level of the signal in Volts you want to generate.
	<b>reference source:</b> is the internal/external setting of the reference voltage for this channel. 0: Do not change the reference source setting (default input). 1: Internal (default setting). 2: External.
	<b>update rate:</b> is the number of updates to generate per second. The default rate is 1000 update/s.
	<b>waveforms:</b> is a 2D array that contains analog output data expressed in volts. You must supply this data. The channel order of the data must be the same channel order specified in channels. You must specify waveforms, where the first (top) dimension is the update number and the second (bottom) dimension is the channel number.

**KI AO update channel**

Writes a single value to the specified analog output channel. If an error occurs, a dialog box appears, giving you the error information.

Table D-14  
**KI AO update channel**








	<b>device:</b> Number of the device (beginning from 1). The utility <b>Device Browser</b> can be used to get the information of current device configuration.
	<b>sub type:</b> is the sub-type of the device you assigned to the Keithley Instruments PXI device during configuration.
	<b>channel:</b> identifies the analog output channel. The default input is channel 0. The valid channel for each Keithley Instruments PXI device is as follows: <b>KPXI-SDAQ-4-500K/KPXI-SDAQ-4-2M/KPXI-DAQ-64-3M/KPXI-DAQ-64-500K/</b> <b>KPXI-DAQ-64-250K/KPXI-DAQ-96-3MKI:</b> 0 or 1 <b>KPXI-AO-4-1M:</b> 0 through 3 <b>KPXI-AO-8-1M:</b> 0 through 7
	<b>high limit:</b> is the highest expected level of the signal in Volts you want to generate.
	<b>low limit:</b> is the lowest expected level of the signal in Volts you want to generate.








Table D-14 (continued)  
**KI AO update channel**

	<b>reference source:</b> is the internal/external setting of the reference voltage for this channel. 0: Do not change the reference source setting (default input). 1: Internal (default setting). 2: External.
	<b>value:</b> contains the value to be written to the specified analog output channel expressed in the physical units of your signal. You must supply this data. All boards require Voltage for the physical unit.

### KI AO update channels

Writes values to each of the specified analog output channels. If an error occurs, a dialog box appears, giving you the error information.

Table D-15  
**KI AO update channels**

	<b>device:</b> Number of the device (beginning from 1). The utility <b>Device Browser</b> can be used to get the information of current device configuration.
	<b>sub type:</b> is the sub-type of the device you assigned to the Keithley Instruments PXI device during configuration.
	<b>channels:</b> Specifies the set of analog output channels you want to use. If x, y, and z refer to channels, you can specify a list of channels by "x,y,z". If x refers to the first channel in a consecutive channel range and y refers to the last channel, you can specify the range by "x:y". See KI AO update channel above for available channels on each module.
	<b>high limit:</b> is the highest expected level of the signal in Volts you want to generate.
	<b>low limit:</b> is the lowest expected level of the signal in Volts you want to generate.
	<b>reference source:</b> is the internal/external setting of the reference voltage for this channel. 0: Do not change the reference source setting (default input). 1: Internal (default setting). 2: External.
	<b>value:</b> is a 1D array that contains the analog output data expressed in the physical units of your signal. You must supply this data. All boards require Voltage for the physical unit.











## Intermediate analog output VIs

### KI AO clear

This VI stops an analog output generation associated with taskID in and releases associated internal resources, including buffers. Before beginning a new signal generation, you must call the KI AO Config VI.

**NOTE** This VI is not supported for Keithley KDIO Series devices.

Table D-16  
**KI AO clear**

	<b>taskID in:</b> identifies the group and the I/O operation.
	<b>error in (no error):</b> describes error conditions occurring before the VI executes. If an error has already occurred, the VI returns the value of the error in cluster in error out.
	<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
	<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
	<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.
	<b>taskID out:</b> has the same value as taskID in.
	<b>error out:</b> contains error information. If the error in cluster indicated an error, the error out cluster contains the same information. Otherwise, error out describes the error status of this VI.
	<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
	<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
	<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.

**KI AO Config**

Configures a buffered analog output operation, including configuring the hardware and allocating a buffer.

**NOTE** This VI is not supported for Keithley KDIO Series devices.

Table D-17  
**KI AO Config**




	<b>interchannel delay:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
	<b>sub type:</b> is the sub-type of the device you assigned to the Keithley Instruments PXI device during configuration.
	<b>limit settings:</b> is an array of clusters. Each array element contains the expected signal limits for the channels specified by the corresponding element of channels. If there are fewer elements in this array than in channels, the VI uses the last array element for the rest of the channels. The default for the limit settings array is an empty array, which means the limit settings keep their default settings.

Table D-17 (continued)  
**KI AO Config**



















		<b>high limit:</b> is the highest scaled data in Volts.
		<b>low limit:</b> is the lowest scaled data in Volts.
		<b>reference source:</b> is the internal/external setting of the reference voltage for this channel. 0: Do not change the reference source setting (default input). 1: Internal (default setting). 2: External.
		<b>device:</b> Number of the device (beginning from 1). The utility <b>Device Browser</b> can be used to get the information of current device configuration.
		<b>channels:</b> specifies the set of analog output channels. channels is an array of strings. If x, y, and z refer to channels, you can specify a list of channels in a single element by separating the individual channels by commas. For example, "x,y,z". If x refers to the first channel in a consecutive channel range and y refers to the last channel, you can specify the range by separating the first and last channels by a colon. For example, "x:y". The valid channel order for data is as follows: <b>KPXI-SDAQ-4-500K/KPXI-SDAQ-4-2M/KPXI-DAQ-64-3M/KPXI-DAQ-64-500K/</b> <b>KPXI-DAQ-64-250K/KPXI-DAQ-96-3M:</b> numbers in <b>channels</b> must be within 0 and 1 <b>KPXI-AO-4-1M:</b> numbers in <b>channels</b> must be within 0 and 3 <b>KPXI-AO-8-1M:</b> numbers in <b>channels</b> must be within 0 and 7
		<b>buffer size:</b> is the number of updates you want the buffer to hold. The default for this parameter is 1000 scans.
		<b>group:</b> is the number, from 0 to 15, that you assign to the specified set of channels. The default input and setting for group is 0. If you only have one update operation for this device, leave this input unwired and use group 0.
		<b>error in (no error):</b> describes error conditions occurring before the VI executes. If an error has already occurred, the VI returns the value of the error in cluster in error out.
		<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
		<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
		<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.
		<b>allocation mode:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
		<b>taskID:</b> identifies the group and the I/O operation.
		<b>number of channels:</b> is the total number of channels in the group.

Table D-17 (continued)

**KI AO Config**

	<b>error out:</b> contains error information. If the error in cluster indicated an error, the error out cluster contains the same information. Otherwise, error out describes the error status of this VI.
	<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
	<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
	<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.

**KI AO start**

Starts a buffered analog output operation. This VI sets the update rate, and then starts the generation.

**NOTE** This VI is not supported for Keithley KDIO Series devices.

Table D-18

**KI AO start**
















	<b>taskID in:</b> identifies the group and the I/O operation.
	<b>update rate:</b> is the number of updates/s to generate. This is equivalent to the update rate per channel. The default for this parameter is 1000 updates/s. If you enter 0, the on-board internal clock is disabled and the external clock is used.
	<b>error in (no error):</b> describes error conditions occurring before the VI executes. If an error has already occurred, the VI returns the value of the error in cluster in error out.
	<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
	<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
	<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.
	<b>number of buffer iterations:</b> is the number of times KIDAQ LabVIEW has to generate the waveform from the output buffer. After generating the buffer the specified number of times, the generation stops. The default value is 1, which means KIDAQ LabVIEW generates the buffer only once. If you use a value of 0, KIDAQ LabVIEW generates the buffer continuously, until you stop the operation with the KI AO Clear VI.
	<b>clock:</b> 0: Do not change the default setting (default input). 1: Update clock 1 (default setting).

Table D-18 (continued)

**KI AO start**

	<b>clock source:</b> specifies the source of the clock. 0: Do not change the clock source setting (default input). 1: Internal (default setting). 6: I/O connector. 7: SSI (RTSI) Connection.
	<b>taskID out:</b> has the same value as taskID in.
	<b>error out:</b> contains error information. If the error in cluster indicated an error, the error out cluster contains the same information. Otherwise, error out describes the error status of this VI.
	<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
	<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
	<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.
	<b>actual update rate:</b> may differ slightly from the requested update rate, depending on the hardware capabilities.

**KI AO wait**

This VI waits until the waveform generation of the task completes before returning.

**NOTE** *This VI is not supported for Keithley KDIO Series devices.*

Table D-19

**KI AO wait**













	<b>taskID in:</b> identifies the group and the I/O operation.
	<b>update rate:</b> is the number of updates/s to generate. This is equivalent to the update rate per channel. The default for this parameter is 1000 updates/s.
	<b>error in (no error):</b> describes error conditions occurring before the VI executes. If an error has already occurred, the VI returns the value of the error in cluster in error out.
	<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
	<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
	<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.
	<b>check every N updates:</b> informs the VI how often to check the status of the task to see if generation completes. This parameter default is to check every 5 updates.



Table D-19 (continued)  
**KI AO wait**

	<b>taskID out:</b> has the same value as taskID in.
	<b>error out:</b> contains error information. If the error in cluster indicated an error, the error out cluster contains the same information. Otherwise, error out describes the error status of this VI.
	<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
	<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
	<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.

**KI AO write**

This VI writes data into the buffer for a buffered analog output operation. The data written into the buffer will then be generated (transferred from the buffer to the DAC) at the update rate specified in KI AO Start.

KI AO Write is a polymorphic VI that you can configure to output the following kinds of data:

- Binary Array
- Scaled Array

**NOTE** This VI is not supported for Keithley KDIO Series devices.

Table D-20  
**KI AO write binary array**







	<b>taskID in:</b> identifies the group and the I/O operation.
	<b>binary data:</b> is a 2D array that contains unscaled analog output data. The channel order of the data must be the same as the channel order you specify in channels. You must specify waveforms, where the first (top) dimension is the update number and the second (bottom) dimension is the channel number. The length of the data array determines the number of updates the VI writes. When no data is wired, this VI is still useful for reporting update progress information.
	<b>time limit in sec:</b> is the time limit for the output operation. The default input is -1.0, which means KIDAQ LabVIEW calculates a time limit based on the value of number of updates to generate and the update rate. If the update rate is unknown, the VI uses 1 second as the time limit. The resolution of the timeout clock is about 55 ms.
	<b>allow regeneration:</b> is not used by Keithley Instruments PXI devices and is ignored.
	<b>error in (no error):</b> describes error conditions occurring before the VI executes. If an error has already occurred, the VI returns the value of the error in cluster in error out.
	<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.

Table D-20 (continued)  
**KI AO write binary array**











	<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
	<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.
	<b>taskID out:</b> has the same value as <i>taskID in</i> .
	<b>number of updates done:</b> is the number of updates the VI has generated; that is, the number of updates the VI has actually transferred from the buffer to the onboard FIFO.
	<b>number of buffers done:</b> is the number of times the VI has generated an entire buffer; that is, the number of times the VI has actually transferred all the data in the buffer to the onboard FIFO.
	<b>generation complete:</b> is TRUE when the generation finishes.
	<b>error out:</b> contains error information. If the error in cluster indicated an error, the error out cluster contains the same information. Otherwise, error out describes the error status of this VI.
	<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
	<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
	<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.

Table D-21  
**KI AO write binary array scaled array**

















	<b>taskID in:</b> Identifies the group and the I/O operation.
	<b>scaled data:</b> is a 2D array that contains analog output data expressed in volts. The channel order of the data must be the same the channel order you specify in channels. You must specify waveforms, where the first (top) dimension is the update number and the second (bottom) dimension is the channel number. The length of the data array determines the number of updates the VI writes. When no data is wired, this VI is still useful for reporting update progress information.
	<b>time limit in sec:</b> is the time limit for the output operation. The default input is -1.0, which means KIDAQ LabVIEW calculates a time limit based on the value of number of updates to generate and the update rate. If the update rate is unknown, the VI uses 1 second as the time limit. The resolution of the timeout clock is about 55 ms.
	<b>allow regeneration:</b> is not used by Keithley Instruments PXI devices and is ignored.

Table D-21 (continued)  
**KI AO write binary array scaled array**

	<b>error in (no error):</b> describes error conditions occurring before the VI executes. If an error has already occurred, the VI returns the value of the error in cluster in error out.	
		<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
		<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
		<b>source:</b> Identifies where an error occurred. The source string is usually the name of the VI that produced the error.
	<b>taskID out:</b> has the same value as <i>taskID in</i> .	
	<b>number of updates done:</b> is the number of updates the VI has generated; that is, the number of updates the VI has actually transferred from the buffer to the onboard FIFO.	
	<b>number of buffers done:</b> is the number of times the VI has generated an entire buffer; that is, the number of times the VI has actually transferred all the data in the buffer to the onboard FIFO.	
		<b>generation complete:</b> is TRUE when the generation finishes.
	<b>error out:</b> contains error information. If the error in cluster indicated an error, the error out cluster contains the same information. Otherwise, error out describes the error status of this VI.	
		<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
		<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
		<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.

## Advanced analog output VIs

### KI AO Trigger and Gate Config

Configures the trigger conditions for analog output operations.

**NOTE** This VI is not supported for Keithley KDIO Series devices.

Table D-22

### KI AO Trigger and Gate Config













	<b>taskID in:</b> identifies the group and the I/O operation.
	<b>error in (no error):</b> describes error conditions occurring before the VI executes. If an error has already occurred, the VI returns the value of the error in cluster in error out.
	<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
	<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
	<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.
	<b>trigger or gate source:</b> specifies the source of trigger signal. 0: No change (default input). 1: None (default setting). 2: External WFDTRIG pin . 3: SSI (RTSI) pin . 5: ATCOUT (the output of the analog trigger circuitry).
	<b>trigger or gate condition:</b> selects a rising or falling edge trigger. 0: No change (default input). 1: None (default setting). 2: Trigger on rising edge. 3: Trigger on falling edge.
	<b>trigger or gate source specification:</b> is not used by Keithley Instruments PXI devices and is ignored.
	<b>additional trig params:</b> cluster contains the following parameters:
	<b>delay:</b> specifies how long the device waits after a trigger occurs before waveform generates. You express delay in seconds. The default input and setting are 0.0s (no delay).
	<b>taskID out:</b> has the same value as <i>taskID in</i> .
	<b>error out:</b> contains error information. If the error in cluster indicated an error, the error out cluster contains the same information. Otherwise, error out describes the error status of this VI.

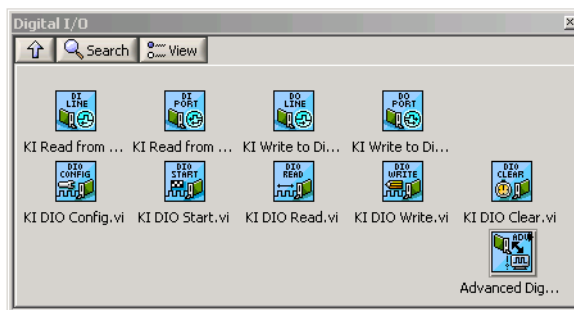
Table D-22 (continued)  
**KI AO Trigger and Gate Config**

	<b>TF</b>	<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
	<b>I32</b>	<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
	<b>abc</b>	<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.
<b>[000]</b>		<b>actual trigger params:</b> cluster may differ slightly from the requested trigger inputs, depending on the hardware capabilities. It contains the following parameters.
	<b>SGL</b>	<b>actual delay:</b> is the delay the VI used.

## Digital I/O VIs

Two classes of Digital I/O VIs are available in the Digital I/O palette: the Easy Digital I/O VIs, Intermediate Digital I/O VIs and Advanced Digital I/O VIs ([Figure D-3](#)).

Figure D-3  
**Digital I/O palette**



## Easy Digital I/O VIs

### KI Read from Digital Line








Reads the logical state of a digital line on a digital port. If an error occurs, a dialog box appears, giving you the error information.

**NOTE** When you call this VI on a digital I/O port that is part of an 8255 PPI and your iteration terminal is left at 0, the 8255 PPI goes through a configuration phase, where all the ports within the same PPI chip get reset to logic low, regardless of the data direction. The data

*direction on other ports, however, is maintained. To avoid this effect, connect a value other than 0 to the iteration terminal once you have configured the desired ports.*

Table D-23

**KI Read from Digital Line**

	<b>device:</b> Number of the device (beginning from 1). The utility <b>Device Browser</b> can be used to get the information of current device configuration.
	<b>sub type:</b> is the sub-type of the device you assigned to the Keithley Instruments PXI device during configuration.
	<p><b>digital channel:</b> is the port number to read.  <b>KPXI-DIO-16-16:</b> 1  <b>KPXI-DIO-48:</b>  0: P1A, 1: P1B, 2: P1C Lower, 3: P1C Upper  4: P2A, 5: P2B, 6: P2C Lower, 7: P2C Upper</p> <p><b>KPXI-RDI-8-16:</b> 1  <b>KPXI-DIO-32-80M:</b> 3 (aux. input port)  <b>KPXI-DIO-32-32:</b> 2 or 3  <b>KPXI-DIO-64-0:</b> 1 or 2  <b>KPXI-DIO-0-64:</b> 3  <b>KPXI-DAQ series devices:</b>  0: P1A, 1: P1B, 2: P1C Lower, 3: P1C Upper</p>
	<b>Line:</b> is the individual port bit or line to be used for I/O.
	<b>Line state:</b> is TRUE for high logic, and FALSE for low logic.
	<b>Port width:</b> is the total width or the number of lines of the port in bits. For example, you can combine two 4-bit ports into an 8-bit port on a KPXI-DIO-48 device by setting port width to 8.
	<b>iteration:</b> When iteration is 0 (default), KIDAQ LabVIEW re-configures the port. If iteration is greater than zero, KIDAQ LabVIEW uses the existing configuration, which improves performance. It can be used to optimize operation when you execute this VI in a loop.

**KI Read from Digital Port**

Reads a digital channel that you configure. If an error occurs, a dialog box appears, giving you the error information.

**NOTE** *When you call this VI on a digital I/O port that is part of an 8255 PPI and your iteration terminal is left at 0, the 8255 PPI goes through a configuration phase, where all the ports within the same PPI chip get reset to logic low, regardless of the data direction. The data direction on other ports, however, is maintained. To avoid this effect, connect a value other than 0 to the iteration terminal once you have configured the desired ports.*

Table D-24

**KI Read from Digital Port**







	<b>device:</b> Number of the device (beginning from 1). The utility <b>Device Browser</b> can be used to get the information of current device configuration.
	<b>sub type:</b> is the sub-type of the device you assigned to the Keithley Instruments PXI device during configuration.

Table D-24 (continued)  
**KI Read from Digital Port**

	<p><b>digital channel:</b> is the port number to read.  <b>KPXI-DIO-16-16:</b> 1</p> <p><b>KPXI-DIO-48:</b>                      0: P1A, 1: P1B, 2: P1C Lower, 3: P1C Upper                      4: P2A, 5: P2B, 6: P2C Lower, 7: P2C Upper</p> <p><b>KPXI-RDI-8-16:</b> 1  <b>KPXI-DIO-32-80M:</b> 3 (aux. input port)  <b>KPXI-DIO-32-32:</b> 2 or 3  <b>KPXI-DIO-64-0:</b> 1 or 2  <b>KPXI-DIO-0-64:</b> 3</p> <p><b>KPXI-DAQ series devices:</b>                      0: P1A, 1: P1B, 2: P1C Lower, 3: P1C Upper</p>
	<p><b>pattern:</b> is the data the VI reads from the digital port.</p>
	<p><b>port width:</b> is the total width or the number of lines of the port in bits. For example, you can combine two 4-bit ports into an 8-bit port on a KPXI-DIO-48 device by setting port width to 8.</p>
	<p><b>iteration:</b> When iteration is 0 (default), KIDAQ LabVIEW re-configures the port. If iteration is greater than zero, KIDAQ LabVIEW uses the existing configuration, which improves performance. It can be used to optimize operation when you execute this VI in a loop.</p>

**KI Write to Digital Line**

Sets the logic state of a digital line on a specified digital port. If an error occurs, a dialog box appears, giving you the error information.

**NOTE** *When you call this VI on a digital I/O port that is part of an 8255 PPI and your iteration terminal is left at 0, the 8255 PPI goes through a configuration phase, where all the ports within the same PPI chip get reset to logic low, regardless of the data direction. The data direction on other ports, however, is maintained. To avoid this effect, connect a value other than 0 to the iteration terminal once you have configured the desired ports.*

Table D-25  
**KI Write to Digital Line**








	<p><b>device:</b> Number of the device (beginning from 1). The utility <b>Device Browser</b> can be used to get the information of current device configuration.</p>
	<p><b>sub type:</b> is the sub-type of the device you assigned to the Keithley Instruments PXI device during configuration.</p>

Table D-25 (continued)  
**KI Write to Digital Line**

	<p><b>digital channel:</b> is the port number to write.</p> <p><b>KPXI-DIO-16-16:</b> 0</p> <p><b>KPXI-DIO-48:</b>  0: P1A, 1: P1B, 2: P1C Lower, 3: P1C Upper  4: P2A, 5: P2B, 6: P2C Lower, 7: P2C Upper</p> <p><b>KPXI-RDI-8-16:</b> 0</p> <p><b>KPXI-DIO-32-80M:</b> 1 (aux. output port)</p> <p><b>KPXI-DIO-32-32:</b> 0 (DO) or 1 (LED)</p> <p><b>KPXI-DIO-64-0:</b> 0 (LED)</p> <p><b>KPXI-DIO-0-64:</b> 0 (DO Low), 1 (DO High)</p> <p><b>KPXI-DIO-0-64:</b> 0 (DO Low), 1 (DO High), 2 (LED)</p> <p><b>KPXI-DAQ series devices:</b>  0: P1A, 1: P1B, 2: P1C Lower, 3: P1C Upper</p>
	<p><b>line:</b> is the individual port bit or line to be used for I/O.</p>
	<p><b>line state:</b> is TRUE for high logic, and FALSE for low logic.</p>
	<p><b>port width:</b> is the total width or the number of lines of the port in bits. For example, you can combine two 4-bit ports into an 8-bit port on a KPXI-DIO-48 device by setting port width to 8.</p>
	<p><b>iteration:</b> When iteration is 0 (default), KIDAQ LabVIEW re-configures the port. If iteration is greater than zero, KIDAQ LabVIEW uses the existing configuration, which improves performance. It can be used to optimize operation when you execute this VI in a loop.</p>

### KI Write to Digital Port

Writes a digital pattern to a digital port. If an error occurs, a dialog box appears, giving you the error information.

**NOTE** *When you call this VI on a digital I/O port that is part of an 8255 PPI when your iteration terminal is left at 0, the 8255 PPI goes through a configuration phase, where all the ports within the same PPI chip get reset to logic low, regardless of the data direction. The data direction on other ports, however, is maintained. To avoid this effect, connect a value other than 0 to the iteration terminal once you have configured the desired ports.*

Table D-26  
**KI Write to Digital Port**







	<p><b>device:</b> Number of the device (beginning from 1). The utility <b>Device Browser</b> can be used to get the information of current device configuration.</p>
	<p><b>sub type:</b> is the sub-type of the device you assigned to the Keithley Instruments PXI device during configuration.</p>



Table D-26 (continued)  
**KI Write to Digital Port**

	<p><b>digital channel:</b> is the port number to write.</p> <p><b>KPXI-DIO-16-16:</b> 0</p> <p><b>KPXI-DIO-48:</b>                  0: P1A, 1: P1B, 2: P1C Lower, 3: P1C Upper                  4: P2A, 5: P2B, 6: P2C Lower, 7: P2C Upper</p> <p><b>KPXI-RDI-8-16:</b> 0</p> <p><b>KPXI-DIO-32-80M:</b> 1 (aux. output port)</p> <p><b>KPXI-DIO-32-32:</b> 0 (DO) or 1 (LED)</p> <p><b>KPXI-DIO-64-0:</b> 0 (LED)</p> <p><b>KPXI-DIO-0-64:</b> 0 (DO Low), 1 (DO High)</p> <p><b>KPXI-DIO-0-64:</b> 0 (DO Low), 1 (DO High), 2 (LED)</p> <p><b>KPXI-DAQ series devices series:</b>                  0: P1A, 1: P1B, 2: P1C Lower, 3: P1C Upper</p>
	<p><b>pattern:</b> is the bit pattern writes to the digital port.</p>
	<p><b>port width:</b> is the total width or the number of lines of the port in bits. For example, you can combine two 4-bit ports into an 8-bit port on a KPXI-DIO-48 device by setting port width to 8. If you are using channel names, port width is not needed and is ignored.</p>
	<p><b>iteration:</b> When iteration is 0 (default), KIDAQ LabVIEW re-configures the port. If iteration is greater than zero, KIDAQ LabVIEW uses the existing configuration, which improves performance. It can be used to optimize operation when you execute this VI in a loop.</p>

## Intermediate Digital I/O VIs

### KI DIO Clear

This VI stops a digital input or output acquisition. Before beginning a new acquisition, you must call the KI DIO Config VI.

**NOTE** This VI is not supported for KPXI-DAQ series devices.

Table D-27  
**KI DIO Clear**











	<p><b>taskID in:</b> identifies the group and the I/O operation.</p>	
	<p><b>error in (no error):</b> describes error conditions occurring before the VI executes. If an error has already occurred, the VI returns the value of the error in cluster in error out.</p>	
		<p><b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.</p>
		<p><b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.</p>

Table D-27 (continued)

**KI DIO Clear**

		<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.
		<b>taskID out:</b> has the same value as <i>taskID in</i> .
		<b>error out:</b> contains error information. If the error in cluster indicated an error, the error out cluster contains the same information. Otherwise, error out describes the error status of this VI.
		<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
		<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
		<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.

**KI DIO Config**

Configures a buffered digital I/O operation, including configuring the hardware and allocating a buffer. The VI only applies to KPXI-DIO-32-80M devices.

**NOTE** This VI is not supported for KPXI-DAQ series devices.

Table D-28

**KI DIO Config**








		<b>device:</b> Number of the device (beginning from 1). The utility <b>Device Browser</b> can be used to get the information of current device configuration.
		<b>sub type:</b> is the sub-type of the device you assigned to the Keithley Instruments PXI device during configuration.
		<b>port list:</b> Specifies the set of digital ports, each of which is 8, 16 or 32 lines wide. The valid ports are as follows: KPXI-DIO-32-80M 0 (digital output port), 2 (digital input port)
		<b>port width:</b> is the total width or the number of lines of the port in bits. port width is only valid for KPXI-DIO-32-80M which supports 8-bit, 16-bit and 32-bit of data acquisition
		<b>group direction:</b> sets the direction for the group. 0: Do not change the group direction setting (default input). 1: Input (default setting). 2: Output.
		<b>error in (no error):</b> describes error conditions occurring before the VI executes. If an error has already occurred, the VI returns the value of the error in cluster in error out.
		<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.

Table D-28 (continued)  
**KI DIO Config**























		<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
		<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.
		<b>number of scans/ updates:</b> specifies how much memory to allocate for the buffer. The default input for number of scans/updates is -1, which means KIDAQ LabVIEW leaves the current setting for number of scans/updates unchanged. The default setting for number of scans/updates is 1000.
		<b>group:</b> is the number the VI assigns to the set of ports, ranging from 0 to 15. The default input and setting for group is 0.
		<b>handshaking mode parameters:</b> affects the handshaking operation of KPXI-DIO-32-80M devices.
		<b>signal mode:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
		<b>edge mode:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
		<b>request polarity:</b> specifies active high or low handshaking request signals. 0: Do not change the request polarity setting (default input). 1: Active low requests (default setting). 2: Active high requests.
		<b>acknowledge polarity:</b> specifies active high or low handshaking acknowledge signals. 0: Do not change the acknowledge polarity setting (default input). 1: Active low acknowledges (default setting). 2: Active high acknowledges.
		<b>acknowledge modify mode:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
		<b>acknowledge modify amount:</b> This input is not used by Keithley Instruments PXI devices and is ignored
		<b>hardware double-buffer mode:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
		<b>terminator:</b> is TRUE if output port terminator is on and is FALSE if output port terminator is off. Terminator affects only KPXI-DIO-32-80M.
		<b>burst handshaking enable:</b> is TRUE if burst handshaking mode is enabled and is FALSE if burst handshaking mode is disabled. burst handshaking enable affects only KPXI-DIO-32-80M.
		<b>fifo control:</b> controls the DO FIFO. This parameter is only valid for KPXI-DIO-32-80M.

Table D-28 (continued)

**KI DIO Config**

		<b>fifo wait enable:</b> TRUE: delay output data until FIFO is not almost empty FALSE: digital output does not wait for FIFO is not almost empty.
		<b>threshold:</b> Is the programmable almost empty threshold of both PORTB FIFO and PORTA FIFO (if output port width is 32).
	<b>taskID:</b> identifies the group and the I/O operation.	
	<b>error out:</b> contains error information. If the error in cluster indicated an error, the error out cluster contains the same information. Otherwise, error out describes the error status of this VI.	
		<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
		<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
		<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.

**KI DIO Read**

Reads data from the internal buffer and returns the data read in pattern.

KI DIO Read is a polymorphic VI that you can configure to output the following kinds of data:

- U8 Array (with port width 8)
- U16 Array (with port width 16)
- U32 Array (with port width 32)

**NOTE** This VI is not supported for KPXI-DAQ series devices.

Table D-29

**KI DIO Read**






















	<b>taskID in:</b> identifies the group and the I/O operation.	
	<b>number of scans to read:</b> is the number of scans to retrieve from buffer. This parameter defaults to -1, which means leaving the number of scans to read setting unchanged. The default setting is equal to the size of the buffer, which you set by KI DIO Config VI. If number of scans to read is 0, you can check the scan backlog to determine how many scans have accumulated. The VI waits until the data is available or the time limit expires.	
	<b>read location:</b> This input is not used by Keithley Instruments PXI devices and is ignored. The starting point for the read is the position where the read mark points to. Initially, the read mark points to the beginning of the acquisition buffer. As you retrieve data from the buffer using this VI, the read mark is incremented to point to the next block of data to be read.	
		<b>read offset:</b> This input is not used by Keithley Instruments PXI devices and is ignored.

Table D-29 (continued)  
**KI DIO Read**

		<b>read mode:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
		<b>error in (no error):</b> describes error conditions occurring before the VI executes. If an error has already occurred, the VI returns the value of the error in cluster in error out.
		<b>Status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
		<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
		<b>Source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.
		<b>time limit in sec:</b> timeout for data read. The default input is -1.0, which means KIDAQ LabVIEW calculates a time limit based on the value of number of scans to read and the scan rate. If the scan rate is unknown, the VI uses 1 second as the time limit. The resolution of the timeout clock is about 55 ms.
		<b>taskID out:</b> has the same value as <i>taskID in</i> .
 or  or 		<b>port data:</b> is a 1D array containing the digital data that the VI obtained from the internal buffer. Each element in this array is an 8-bit, 16-bit or 32-bit unsigned integer that represents a single port data.
		<b>error out:</b> contains error information. If the error in cluster indicated an error, the error out cluster contains the same information. Otherwise, error out describes the error status of this VI.
		<b>Status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
		<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
		<b>Source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.
		<b>scan backlog:</b> is the amount of data in the buffer that remains unread after this VI completes.
		<b>number read:</b> is the number of scans returned.
		<b>retrieval complete:</b> is TRUE when the total number of the scans you specified in the KI DIO Start VI has been read.

**KI DIO Start**

Starts a buffered digital I/O operation.

**NOTE** This VI is not supported for KPXI-DAQ series devices.

Table D-30

**KI DIO Start**

















	<b>taskID in:</b> identifies the group and the I/O operation.
	<b>number of scans /updates to acquire or generate:</b> is the total number of scans to acquire or generate. With the default input -1, the device acquires or generates exactly one buffer of data. The buffer size input to the KI DIO Config VI determines the size of the buffer. If number of scans/updates to acquire or generate is 0, the device acquires or generates data continuously until you stop the operation.
	<b>trigger type:</b> specifies the type of trigger. 0: Do not change (default input). 1: Start trigger. KIDAQ LabVIEW waits trigger signal to start DIO operation.
	<b>trigger mode:</b> sets the trigger on or off. 0: Do not change (default input). 1: Off (default setting). 2: On.
	<b>trigger condition:</b> specifies when the digital operation triggers. 0: Do not change (default input). 1: Trigger on rising edge (default setting). 2: Trigger on falling edge.
	<b>error in (no error):</b> describes error conditions occurring before the VI executes. If an error has already occurred, the VI returns the value of the error in cluster in error out.
	<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
	<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
	<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.
	<b>handshake source:</b> determines the source of the conditions that perform a data transfer. 0: Do not change the handshake source setting (default input). 1: Internal clock. 2: I/O connector (default setting). When handshake source is 1, the clock frequency control determines the clock rate. When handshake source is 2, you must connect the handshake signal to the proper line on the I/O connector.
	<b>clock frequency:</b> is the rate to which you want to handshake the data. This parameter is expressed in scans/s or updates/s. This parameter defaults to -1.0. The default setting is undefined.
	<b>taskID out:</b> has the same value as <b>taskID in</b> .

Table D-30 (continued)

**KI DIO Start**

	<b>error out:</b> contains error information. If the error in cluster indicated an error, the error out cluster contains the same information. Otherwise, error out describes the error status of this VI.	
		<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
		<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
		<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.

**KI DIO Write**

Writes digital output data to the internal transfer buffer. You can call the KI DIO Write VI after the transfer begins to retrieve the output status information.

**NOTE** This VI is not supported for KPXI-DAQ series devices.

Table D-31

**KI DIO Write**




















	<b>taskID in:</b> identifies the group and the I/O operation.	
 or  or 	<b>digital data:</b> is a 1D array containing digital output data. Each element in this array is an 8-bit, 16-bit and 32-bit unsigned integer that represents a single port data. If you call this VI with an empty array, you can examine buffer iterations and generation complete to retrieve the output progressing information.	
	<b>error in (no error):</b> describes error conditions occurring before the VI executes. If an error has already occurred, the VI returns the value of the error in cluster in error out.	
		<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
		<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
		<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.
	<b>time limit in sec:</b> timeout for data write. The default input is -1.0, which means KIDAQ LabVIEW calculates a time limit based on the value of number of updates and the scan rate. If the scan rate is unknown, the VI uses 1 second as the time limit. The resolution of the timeout clock is about 55 ms.	
	<b>write location:</b> Determines where the write begins. Contains the following parameters.	

Table D-31 (continued)

**KI DIO Write**

	<b>write offset:</b> The VI adds the value of write offset to the write mark to determine where the write begins. The default input is -1, which means leaving the write offset setting unchanged. This parameter defaults to a setting of 0.
	<b>write mode:</b> Setting write mode to 2 moves the write mark to the beginning of the buffer before the VI adds write offset to the write mark. 0: Do not change the write mode setting (default input). 1: Write at the write mark plus the write offset (default setting). 2: Write at the beginning of the buffer plus the write offset.
	<b>taskID out:</b> has the same value as <i>taskID in</i> .
	<b>error out:</b> contains error information. If the error in cluster indicated an error, the error out cluster contains the same information. Otherwise, error out describes the error status of this VI.
	<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
	<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
	<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.
	<b>buffer iterations:</b> indicates the current number of complete iterations of the buffer.
	<b>generation complete:</b> is TRUE when the number of updates to generate has finished.



## Advanced Digital I/O VIs

### KI DIO Port Config

Configures a digital channel. You can use the task ID that this VI returns only in digital port VIs.

Table D-32

### KI DIO Port Config
















	<b>device:</b> Number of the device (beginning from 1). The utility <b>Device Browser</b> can be used to get the information of current device configuration.
	<b>sub type:</b> is the sub-type of the device you assigned to the Keithley Instruments PXI device during configuration.
	<p><b>digital channel:</b> is the port number to write.</p> <p><b>KPXI-DIO-16-16:</b> 0</p> <p><b>KPXI-DIO-48:</b> 0: P1A, 1: P1B, 2: P1C Lower, 3: P1C Upper 4: P2A, 5: P2B, 6: P2C Lower, 7: P2C Upper</p> <p><b>KPXI-RDI-8-16:</b> 0</p> <p><b>KPXI-DIO-32-80M:</b> 1 (auxiliary output port)</p> <p><b>KPXI-DIO-32-32:</b> 0 (DO) or 1 (LED)</p> <p><b>KPXI-DIO-64-0:</b> 0 (LED)</p> <p><b>KPXI-DIO-0-64:</b> 0 (DO Low), 1 (DO High)</p> <p><b>KPXI-DIO-0-64:</b> 0 (DO Low), 1 (DO High), 2 (LED)</p> <p><b>KPXI-DAQ series devices:</b> 0: P1A, 1: P1B, 2: P1C Lower, 3: P1C Upper</p>
	<b>port width:</b> is the total width or the number of lines of the port in bits. port width is only valid for KPXI-DIO-32-80M which supports 8-bit, 16-bit and 32-bit of data acquisition
	<b>error in (no error):</b> describes error conditions occurring before the VI executes. If an error has already occurred, the VI returns the value of the error in cluster in error out.
	<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
	<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
	<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.
	<b>line direction map:</b> specifies the direction of each line in the port. If a bit is 0 in the line map, the line is an input line. If a bit is 1, the line is an output line. Set line direction map to -1 to make all the lines in a port output lines. Set line direction map to 0 to make all the lines in a port input lines. Port C (e.g. P1C, P2C, etc.) are the only ports on which you can configure lines for different directions. The least significant bit in the line map corresponds to line 0 in the port. The line direction map parameter defaults to 0.
	<b>wired OR map:</b> is not used and ignored.

Table D-32 (continued)  
**KI DIO Port Config**

	<b>taskID out:</b> uniquely identifies the digital group. Use this value as the task ID to refer to this group in subsequent digital port VIs.
	<b>error out:</b> contains error information. If the error in cluster indicated an error, the error out cluster contains the same information. Otherwise, error out describes the error status of this VI.
	<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
	<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
	<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.

## Counter VIs

Six Counter VIs are contained in the Counter palette.

### Easy Counter VIs

#### KI Count Events or Time

Configures one or two counters to count external events. An external event is a high or low signal transition on the specified **GPTCn\_SRC** pin of the counter.

**NOTE** This VI is not supported for Keithley KDIO series devices.

Table D-33  
**KI Count Events or Time**












	<b>device:</b> Number of the device (beginning from 1). The utility <b>Device Browser</b> can be used to get the information of current device configuration.
	<b>sub type:</b> is the sub-type of the device you assigned to the Keithley Instruments PXI device during configuration.
	<b>counter:</b> is an array of strings that specifies the counter(s) the VI controls.
	<b>counter size:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
	<b>start/restart:</b> is TRUE to configure and start the counter(s).
	<b>stop:</b> is TRUE to stop the counter(s).
	<b>source edge:</b> is the edge of the counter clock signal. 0: Count on low to high transition. 1: Count on high to low transition.

Table D-33 (continued)  
**KI Count Events or Time**

	<b>count:</b> is the value of the counter at the time it is read. If there are two counters assigned to the task ID, the value of the higher order counter is multiplied by 10000 hex, shifting it to the left 16 bits. The higher order counter is then added to the value of the lower counter.
	<b>seconds till overflow:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
	<b>seconds since last call:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
	<b>seconds since start:</b> This input is not used by Keithley Instruments PXI devices and is ignored.

**KI Generate Delayed Pulse**

Configures and starts a counter to generate a single pulse with the specified delay and pulse-width on the counter **GPTCn\_OUT** pin. A single pulse consists of a delay phase (phase 1), followed by a pulse phase (phase 2), and then returns to the phase 1 level.

**NOTE** This VI is not supported for Keithley KDIO series devices.

Table D-34  
**KI Generate Delayed Pulse**







	<b>device:</b> Number of the device (beginning from 1). The utility <b>Device Browser</b> can be used to get the information of current device configuration.
	<b>sub type:</b> is the sub-type of the device you assigned to the Keithley Instruments PXI device during configuration.
	<b>counter:</b> is an array of strings that specifies the counter(s) the VI controls.
	<b>pulse delays (s or cycles):</b> is the desired duration of the first phase of the pulse, phase 1. The unit is seconds if timebase source is 0 (internal) and cycles if timebase source is 1 (external). If pulse delay is 0.0 and timebase source is 0, the VI selects a minimum delay of three cycles of the timebase used.
	<b>pulse-width (s or cycles):</b> is the desired duration of the second phase of the pulse, phase 2. The unit is seconds if timebase source is 0 (internal) and cycles if timebase source is 1 (external). If pulse-width is 0.0 and timebase source is 0, the VI selects a minimum width of three cycles of the timebase used.
	<b>timebase source:</b> is the clock source. Timebase source is 0 to use an internal signal and 1 to use an external signal for the timebase.

Table D-34 (continued)

**KI Generate Delayed Pulse**

<b>U16</b>	<p><b>gate mode:</b> specifies how the counter <b>GPTCn_GATE</b> signal is used.</p> <p>0: Ungated/software start: ignore the gate source and start when the VI is called (default).</p> <p>1: Count while the gate signal is TTL high.</p> <p>2: Count while the gate signal is TTL low.</p> <p>3: Start counting on the rising edge of the TTL gate signal.</p> <p>4: Start counting on the falling edge of the TTL gate signal.</p> <p>5: Restart counting on each rising edge of the TTL gate signal.</p> <p>6: Restart counting on each falling edge of the TTL gate signal.</p> <p>Use gate mode 3 or 4 to generate one delayed pulse on the first gate edge after starting. Use gate mode 5 or 6 to generate a delayed pulse for each gate edge (i.e., retriggerable one-shot behavior).</p>
<b>U16</b>	<p><b>pulse polarity:</b> is the polarity of the second phase (phase 2) of the pulse.</p> <p>0: High pulse: phase 1 (the delay) is a low TTL level and phase 2 is a high level (default).</p> <p>1: Low pulse: phase 1 is a high TTL level and phase 2 is a low level.</p>
<b>U32</b>	<p><b>taskID of counter out:</b> is the task ID of the specified counter, which generates the delayed pulse.</p>
<b>DBL</b>	<p><b>actual delay (s or cycles):</b> is the achieved delay. It may differ from the desired delay because the hardware has limited resolution and range.</p>
	<p><b>actual width (s or cycles):</b> is the achieved pulse-width. It may differ from the desired width because the hardware has limited resolution and range.</p>

**KI Generate Pulse-Train**

Configures the specified counter to generate a continuous pulse-train on the GPTCn\_OUT pin. The signal has the prescribed frequency, duty cycle, and polarity. Each cycle of the pulse-train consist of a delay phase (phase 1) followed by a pulse phase (phase 2).

**NOTE** This VI is not supported for Keithley DIO series devices.











Table D-35

**KI Generate Pulse-Train**

<b>I16</b>	<p><b>device:</b> Number of the device (beginning from 1). The utility <b>Device Browser</b> can be used to get the information of current device configuration.</p>
<b>U16</b>	<p><b>sub type:</b> is the sub-type of the device you assigned to the Keithley Instruments PXI device during configuration.</p>
<b>[abc]</b>	<p><b>counter:</b> is an array of strings that specifies the counter(s) the VI controls.</p>
<b>I32</b>	<p><b>number of pulses:</b> is the number of pulses you want in the pulse-train. If the value is 0 (default), the VI generates a continuous pulse-train.</p>
<b>DBL</b>	<p><b>frequency:</b> (Hz) is the desired repetition rate of the pulse-train.</p>
	<p><b>duty cycle:</b> is the desired ratio of the durations of phase 2 (phase two) of the pulse to the period of one cycle (1/frequency); default is 0.5. If duty cycle is 0.0 or 1.0, the VI computes the closest achievable duty cycle using a minimum period of three timebase cycles. A duty cycle very close to 0.0 or 1.0 may not be possible.</p>

Table D-35 (continued)

**KI Generate Pulse-Train**

	<b>timebase:</b> is the frequency of the clock. If the value of timebase is 0 or 10000000, internal signal is used; otherwise, an external signal is used.
	<b>gate mode:</b> specifies how the counter <i>GPTCn_GATE</i> signal is used. 0: ungated/software start: ignore the gate source and start when the VI is called. (default). 1: Count while the gate signal is TTL high. 2: Count while the gate signal is TTL low. 3: start (continuous) pulse-train on the rising edge of the TTL gate signal. 4: start (continuous) pulse-train on the falling edge of the TTL gate signal. If number of pulses is 0 (continuous pulse-train), gate mode 3 or 4 generates one pulse per gate edge, which is the behavior of a retriggeable one shot. If number of pulses –1, gate mode 3 or 4 generates a continuous pulse-train.
	<b>pulse polarity:</b> is the polarity of the second phase (phase 2) of the pulse. 0: High pulse: phase 1 (the delay) is a low TTL level and phase 2 is a high level (default). 1: Low pulse: phase 1 is a high TTL level and phase 2 is a low level.
	<b>taskID of counter out:</b> is the task ID of the specified counter, which generates the pulse train.
	<b>taskID of counter -1 out:</b> this output is not used by Keithley Instruments PXI devices and is ignored.
	<b>actual parameters out:</b> is a cluster of lesser parameters. These parameters may differ from the desired parameters because the hardware has limited resolution and range.
	<b>frequency:</b> (Hz) is the achieved frequency.
	<b>duty cycle:</b> is the achieved duty cycled.
	<b>pulse delay:</b> is the achieved minimum delay to the gating pulse.
	<b>pulse-width:</b> is the achieved width of the gating pulse.

**KI Measure Pulse-Width or Period**

Measures the pulse-width (length of time a signal is high or low) or period (length of time between adjacent rising or falling edges) of a TTL signal connected to the counter *GPTCn\_GATE* pin. The method used gates an internal timebase clock with the signal being measured. This VI is useful in measuring the period or frequency (1/period) of relatively low frequency signals, when many timebase cycles occur during the gate.

**NOTE** This VI is not supported for Keithley KDIO Series devices.

Table D-36

**KI Measure Pulse-Width or Period**












	<b>device:</b> Number of the device (beginning from 1). The utility <b>Device Browser</b> can be used to get the information of current device configuration.
---	---

Table D-36 (continued)

**KI Measure Pulse-Width or Period**

	<b>sub type:</b> is the sub-type of the device you assigned to the Keithley Instruments PXI device during configuration.
	<b>counter:</b> is an array of strings that specifies the counter(s) the VI controls.
	<b>type of measurement:</b> identifies the type of pulse-width or period measurement to make. The following illustration demonstrates the various values for type of measurement. 0: Measure high pulse-width from rising to falling edge. 1: Measure low pulse-width from falling to rising edge. 2: Measure period between adjacent rising edges. (default) 3: Measure period between adjacent falling edges
	<b>timebase:</b> is the frequency of the clock. If the value of timebase is 0 or 10000000, internal signal is used; otherwise, an external signal is used.
	<b>pulse-width/period (s) out:</b> is the measured pulse-width or period; it equals count/timebase and may be valid or invalid.
	<b>time limit in sec:</b> is the period to wait for a valid measurement. If time limit is -1.0 (default), the time limit is set to five seconds or four times the range of the counter at the selected timebase ( $4 \times 65,536 / \text{timebase}$ ) in seconds.
	<b>valid?:</b> is TRUE if counter has not underflowed (if <b>count ?4</b> ) or overflowed.
	<b>count:</b> is the value of the counter at the time it is read. For best accuracy, choose a timebase frequency that maximizes the count without overflowing it. If there are two counters assigned to the task ID, the value of the higher order counter is multiplied by 10000 hex, shifting it to the left 16 bits. The higher order counter is then added to the value of the lower counter.
	<b>counter overflow?:</b> is TRUE if counter reaches TC. Overflow does not produce an error.
	<b>timeout:</b> is TRUE if a valid reading is not within the prescribed or computed time limit. The timeout parameter does not produce an error.

**Intermediate Counter VIs****KI Continuous Pulse Generator Config**

Configures a counter to generate a continuous TTL pulse-train on its GPTCn\_OUT pin.

**NOTE** This VI is not supported for Keithley KDIO series devices.

Table D-37

**KI Continuous Pulse Generator Config**


	<b>device:</b> Number of the device (beginning from 1). The utility <b>Device Browser</b> can be used to get the information of current device configuration.
---	---

Table D-37 (continued)  
**KI Continuous Pulse Generator Config**



















	<b>sub type:</b> is the sub-type of the device you assigned to the Keithley Instruments PXI device during configuration.
	<b>counter:</b> is the counter this VI controls.
	<b>error in (no error):</b> describes error conditions occurring before the VI executes. If an error has already occurred, the VI returns the value of the error in cluster in error out.
	<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
	<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
	<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.
	<b>frequency:</b> (Hz) is the desired repetition rate of the pulse-train.
	<b>duty cycle:</b> is the desired ratio of the durations of phase 2 (phase two) of the pulse to the period of one cycle (1/frequency); default is 0.5. If duty cycle is 0.0 or 1.0, the VI computes the closest achievable duty cycle using a minimum period of three timebase cycles. A duty cycle very close to 0.0 or 1.0 may not be possible.
	<b>timebase:</b> is the frequency of the clock. If the value of timebase is 0 or 10000000 (10MHz), internal signal is used; otherwise, an external signal is used.
	<b>gate mode:</b> specifies how the counter <i>GPTCn_GATE</i> signal is used. 0: Ungated/software start: ignore the gate source and start when Counter Start VI is called (default). 1: Count while the gate signal is TTL high after the Counter Start VI is called. 2: Count while the gate signal is TTL low after the Counter Start VI is called. 3: Start counting on the rising edge of the TTL gate signal after the Counter Start VI is called. 4: Start counting on the falling edge of the TTL gate signal after the Counter Start VI is called. If gate mode is 3 or 4, the counter generates a single pulse on each edge.
	<b>pulse polarity:</b> is the polarity of the second phase (phase 2) of the pulse. 0: High pulse: phase 1 (the delay) is a low TTL level and phase 2 is a high level (default). 1: Low pulse: phase 1 is a high TTL level and phase 2 is a low level.
	<b>taskID:</b> is the task ID of the specified counter, which generates the pulse train.
	<b>error out:</b> contains error information. If the error in cluster indicated an error, the error out cluster contains the same information. Otherwise, error out describes the error status of this VI.
	<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
	<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.

Table D-37 (continued)

**KI Continuous Pulse Generator Config**

		<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.
		<b>actual frequency:</b> (Hz) is the achieved frequency. It may differ from the desired frequency because the hardware has limited resolution and range.
		<b>actual duty cycle:</b> is the achieved duty cycled. It may differ from the desired duty cycle because the hardware has limited resolution and range.

**KI Counter Divider Config**

Configures the specified counter to divide a signal on the counter GP\_TC\_CLK pin or on an internal timebase signal using a count value called the timebase divisor. The result is that the signal on the counter GP\_TC\_OUT pin is equal to the frequency of the input signal/timebase divisor.

This VI is not supported for Keithley KDAQ series devices.

Table D-38

**KI Counter Divider Config**




















	<b>device:</b> Number of the device (beginning from 1). The utility <b>Device Browser</b> can be used to get the information of current device configuration.
	<b>sub type:</b> is the sub-type of the device you assigned to the Keithley Instruments PXI device during configuration.
	<b>counter:</b> is the counter this VI controls.
	<b>gate mode:</b> specifies how the signal on the counter's GATE pin is used. 0: Ungated/software start: ignore the gate source and start when KI Counter Start VI is called (default). 1: Count while the gate signal is TTL high after the KI Counter Start VI is called.
	<b>source edge:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
	<b>output:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
	<b>updown source:</b> specifies how the signal on the counter's UPDN pin is used. 0: software control: ignore the UPDN source and control by <b>updownctrl</b> (default). 1: hardware control.
	<b>updown control:</b> specifies the specified counter to count down or count up if <b>updown</b> source is configured to be software control. 0: count down (default). 1: count up.



Table D-38 (continued)  
**KI Counter Divider Config**

	<b>timebase:</b> (Hz) is set to the frequency of the internal signal whose cycles are counted, or is set to <=0.0 (default) to count the rising edges of the signal on the counter GP_TC_CLK pin.
	<b>timebase divisor:</b> is the count down or divide value. For example, if the input frequency is 24000000 Hz, timebase divisor is 240000, and the output is pulsed, the frequency of the counter's GP_TC_OUT signal is 100 Hz.
	<b>error in (no error):</b> describes error conditions occurring before the VI executes. If an error has already occurred, the VI returns the value of the error in cluster in error out.
	<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
	<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
	<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.
	<b>taskID:</b> identifies the group and the I/O operation.
	<b>error out:</b> contains error information. If the error in cluster indicated an error, the error out cluster contains the same information. Otherwise, error out describes the error status of this VI.
	<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
	<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
	<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.

**KI Counter Read**

Reads the counter or counters identified by task ID.

**NOTE** This VI is designed to read general purpose counter of Keithley KDAQ series devices.

Table D-39  
**KI Counter Read**














	<b>taskID in:</b> identifies the group and the I/O operation.
	<b>error in (no error):</b> describes error conditions occurring before the VI executes. If an error has already occurred, the VI returns the value of the error in cluster in error out.
	<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.

Table D-39 (continued)

**KI Counter Read**

	<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
	<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.
	<b>counter list:</b> is the set of counters to read. Use this array only to read a subset of counters identified by task ID; otherwise, leave it empty. This input is only valid for KPXI-DAQ series devices.
	<b>taskID out:</b> has the same value as <i>taskID</i> in.
	<b>count:</b> is the value of the counter at the time it is read. If there are two counters assigned to the task ID, the value of the higher order counter is shifted to 16 bits to scale it, and then it is added to the value of the lower counter.
	<b>error out:</b> contains error information. If the error in cluster indicated an error, the error out cluster contains the same information. Otherwise, error out describes the error status of this VI.
	<b>Status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
	<b>Code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
	<b>Source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.
	<b>overflow:</b> This input is not used by Keithley Instruments PXI devices and is ignored.

**KI Counter Start**

Starts the counters identified by task ID. This applies only to Keithley KDAQ series devices.

Table D-40

**KI Counter Start**












	<b>taskID in:</b> identifies the group and the I/O operation.
	<b>error in (no error):</b> describes error conditions occurring before the VI executes. If an error has already occurred, the VI returns the value of the error in cluster in error out.
	<b>Status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
	<b>Code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.

Table D-40 (continued)

**KI Counter Start**

	<b>Source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.
	<b>counter list:</b> is the set of counters to read. Use this array only to read a subset of counters identified by task ID; otherwise, leave it empty. This input is only valid for Keithley KDAQ series devices.
	<b>taskID out:</b> has the same value as <i>taskID</i> in.
	<b>error out:</b> contains error information. If the error in cluster indicated an error, the error out cluster contains the same information. Otherwise, error out describes the error status of this VI.
	<b>Status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
	<b>Code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
	<b>Source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.

**KI Counter Stop**

Stops a count operation immediately or conditionally on an input error. This applies only to Keithley KDAQ series devices.

Table D-41

**KI Counter Stop**












	<b>taskID in:</b> identifies the group and the I/O operation.
	<b>error in (no error):</b> describes error conditions occurring before the VI executes. If an error has already occurred, the VI returns the value of the error in cluster in error out.
	<b>Status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
	<b>Code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
	<b>Source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.
	<b>stop when:</b> This input is not used by Keithley Instruments PXI devices and is ignored.

Table D-41 (continued)

**KI Counter Stop**

	<b>taskID out:</b> has the same value as <i>taskID in</i> .
	<b>error out:</b> contains error information. If the error in cluster indicated an error, the error out cluster contains the same information. Otherwise, error out describes the error status of this VI.
	<b>Status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
	<b>Code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
	<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.

**KI Delayed Pulse Generator Config**

Configures a counter to generate a single pulse with the specified delay and pulse-width on the counter GPTCn\_OUT pin.

**NOTE** This VI is not supported for Keithley KDIO Series devices.

Table D-42

**KI Delayed Pulse Generator Config**




















	<b>device:</b> Number of the device (beginning from 1). The utility <b>Device Browser</b> can be used to get the information of current device configuration.
	<b>sub type:</b> is the sub-type of the device you assigned to the Keithley Instruments PXI device during configuration.
	<b>counter:</b> is an array of strings that specifies the counter(s) the VI controls.
	<b>pulse delays (s or cycles):</b> is the desired duration of the first phase of the pulse, phase 1. The unit is seconds if timebase source is 0 (internal) and cycles if timebase source is 1 (external). If pulse delay is 0.0 and timebase source is 0, the VI selects a minimum delay of three cycles of the timebase used.
	<b>pulse-width (s or cycles):</b> is the desired duration of the second phase of the pulse, phase 2. The unit is seconds if timebase source is 0 (internal) and cycles if timebase source is 1 (external). If pulse-width is 0.0 and timebase source is 0, the VI selects a minimum width of three cycles of the timebase used.
	<b>timebase source:</b> is the clock source. Timebase source is 0 to use an internal signal and 1 to use an external signal (from <i>GPTCn_SRC</i> pin) for the timebase.

Table D-42 (continued)  
**KI Delayed Pulse Generator Config**

	<p><b>gate mode:</b> specifies how the counter <i>GPTCn_GATE</i> signal is used.</p> <p>0: Ungated/software start: ignore the gate source and start when Counter Start VI is called (default)..</p> <p>1: Count while the gate signal is TTL high after the Counter Start VI is called.</p> <p>2: Count while the gate signal is TTL low after the Counter Start VI is called.</p> <p>3: Start counting on the rising edge of the TTL gate signal after the Counter Start VI is called.</p> <p>4: Start counting on the falling edge of the TTL gate signal after the Counter Start VI is called.</p> <p>Use gate mode 3 or 4 to generate one delayed pulse on the first gate edge after starting.</p>
	<p><b>pulse polarity:</b> is the polarity of the second phase (phase 2) of the pulse.</p> <p>0: High pulse: phase 1 (the delay) is a low TTL level and phase 2 is a high level (default).</p> <p>1: Low pulse: phase 1 is a high TTL level and phase 2 is a low level.</p>
	<p><b>error in (no error):</b> describes error conditions occurring before the VI executes. If an error has already occurred, the VI returns the value of the error in cluster in error out.</p>
	<p><b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.</p>
	<p><b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.</p>
	<p><b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.</p>
	<p><b>taskID:</b> identifies the group and the I/O operation.</p>
	<p><b>error out:</b> contains error information. If the error in cluster indicated an error, the error out cluster contains the same information. Otherwise, error out describes the error status of this VI.</p>
	<p><b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.</p>
	<p><b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.</p>
	<p><b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.</p>
	<p><b>actual delay (s or cycles):</b> is the achieved delay. It may differ from the desired delay because the hardware has limited resolution and range.</p>
	<p><b>actual width (s or cycles):</b> is the achieved pulse-width. It may differ from the desired width because the hardware has limited resolution and range.</p>

### KI Down Counter or Divider Config

Configures the specified counter to count down or divide a signal on the counter **GPTCn\_SRC** pin or on an internal timebase signal using a count value called the timebase divisor. The result is that the signal on the counter **GPTCn\_OUT** pin is equal to the frequency of the input signal/timebase divisor.

**NOTE** This VI is not supported for Keithley KDIO Series devices.

Table D-43

### KI Down Counter or Divider Config


















	<b>device:</b> Number of the device (beginning from 1). The utility <b>Device Browser</b> can be used to get the information of current device configuration.
	<b>sub type:</b> is the sub-type of the device you assigned to the Keithley Instruments PXI device during configuration.
	<b>counter:</b> is an array of strings that specifies the counter(s) the VI controls.
	<b>timebase:</b> (Hz) is set to the frequency of the internal signal whose cycles are counted, or is set to <=0.0 (default) to count the rising edges of the signal on the counter <b>GPTCn_SRC</b> pin.
	<b>timebase divisor:</b> is the count down or divide value. For example, if the input frequency is 10000000 Hz, timebase divisor is 100000, and the output is pulsed, the frequency of the counter's <b>GPTCn_OUT</b> signal is 100 Hz.
	<b>gate mode:</b> specifies how the counter <b>GPTCn_GATE</b> signal is used. 0: Ungated/software start: ignore the gate source and start when Counter Start VI is called (default). 1: Count while the gate signal is TTL high after the Counter Start VI is called. 2: Count while the gate signal is TTL low after the Counter Start VI is called. 3: Start counting on the rising edge of the TTL gate signal after the Counter Start VI is called. 4: Start counting on the falling edge of the TTL gate signal after the Counter Start VI is called.
	<b>source edge:</b> is the edge of the counter clock signal. 0: Count on low to high transition. 1: Count on high to low transition.
	<b>output:</b> is the behavior of the output signal when counter reaches TC. 0: High pulse lasting one cycle of the source or timebase signal (default). 1: Low pulse lasting one cycle of the source or timebase signal. 2: High toggle lasting until the next TC. 3: Low toggle lasting until the next TC. The effect of output modes 0 and 1 is to divide-down the source of timebase frequency by the timebase divisor. The effect of output modes 2 and 3 is to divide the frequency by twice the timebase divisor.
	<b>error in (no error):</b> describes error conditions occurring before the VI executes. If an error has already occurred, the VI returns the value of the error in cluster in error out.
	<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
	<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.

Table D-43 (continued)  
**KI Down Counter or Divider Config**

		<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.
		<b>taskID:</b> identifies the group and the I/O operation.
		<b>error out:</b> contains error information. If the error in cluster indicated an error, the error out cluster contains the same information. Otherwise, error out describes the error status of this VI.
		<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
		<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
		<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.

**KI Event or Time Counter Config**

Configures one or two counters to count external events. An external event is a high or low signal transition on the specified **GPTCn\_SRC** pin of the counter.

**NOTE** This VI is not supported for Keithley KDIO Series devices.

Table D-44  
**KI Event or Time Counter Config**













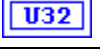




	<b>device:</b> Number of the device (beginning from 1). The utility <b>Device Browser</b> can be used to get the information of current device configuration.
	<b>sub type:</b> is the sub-type of the device you assigned to the Keithley Instruments PXI device during configuration.
	<b>counter:</b> is an array of strings that specifies the counter(s) the VI controls.
	<b>count limit:</b> this input is not used by Keithley Instruments PXI devices and is ignored.
	<p><b>gate mode:</b> specifies how the counter <b>GPTCn_GATE</b> signal is used.</p> <p>0: Ungated/software start: ignore the gate source and start when Counter Start VI is called (default).</p> <p>1: Count while the gate signal is TTL high after the Counter Start VI is called.</p> <p>2: Count while the gate signal is TTL low after the Counter Start VI is called.</p> <p>3: Start counting on the rising edge of the TTL gate signal after the Counter Start VI is called.</p> <p>4: Start counting on the falling edge of the TTL gate signal after the Counter Start VI is called.</p>
	<b>counter size:</b> is not used by Keithley Instruments PXI devices and is ignored.

Table D-44 (continued)

**KI Event or Time Counter Config**

	<b>error in (no error):</b> describes error conditions occurring before the VI executes. If an error has already occurred, the VI returns the value of the error in cluster in error out.	
		<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
		<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
		<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.
	<b>source edge:</b> is the edge of the counter clock signal. 0: Count on low to high transition. 1: Count on high to low transition.	
	<b>event source/timebase:</b> (Hz) is set to the frequency of the internal signal whose cycles are counted, or is set to <=0.0 (default) to count the rising edges of the signal on the counter <i>GPTCn_SRC</i> pin.	
	<b>taskID:</b> identifies the group and the I/O operation.	
	<b>error out:</b> contains error information. If the error in cluster indicated an error, the error out cluster contains the same information. Otherwise, error out describes the error status of this VI.	
		<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
		<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
		<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.

















### KI Pulse-Width or Period Measurement Config

Configures the specified counter to measure the pulse-width or period of a TTL signal connected to its *GPTCn\_GATE* pin.

**NOTE** This VI is not supported for Keithley KDIO Series devices.

Table D-45

#### KI Pulse-Width or Period Measurement Config

	<b>device:</b> Number of the device (beginning from 1). The utility <b>Device Browser</b> can be used to get the information of current device configuration.
	<b>sub type:</b> is the sub-type of the device you assigned to the Keithley Instruments PXI device during configuration.
	<b>counter:</b> is an array of strings that specifies the counter(s) the VI controls.
	<b>timebase:</b> (Hz) is set to the frequency of the internal signal whose cycles are counted, or is set to <=0.0 (default) to count the rising edges of the signal on the counter <i>GPTCn_SRC</i> pin.
	<b>type of measurement:</b> identifies the type of pulse-width or period measurement to make. The following illustration demonstrates the various values for type of measurement. 0: Measure high pulse-width from rising to falling edge. 1: Measure low pulse-width from falling to rising edge. 2: Measure period between adjacent rising edges. (default) 3: Measure period between adjacent falling edges
	<b>error in (no error):</b> describes error conditions occurring before the VI executes. If an error has already occurred, the VI returns the value of the error in cluster in error out.
	<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
	<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
	<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.
	<b>taskID:</b> identifies the group and the I/O operation.
	<b>error out:</b> contains error information. If the error in cluster indicated an error, the error out cluster contains the same information. Otherwise, error out describes the error status of this VI.
	<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
	<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
	<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.

## KI UpDown Counter Config

Configures one counter to count edges in the signal on the specified counter's SOURCE pin or the number of cycles of a specified internal timebase signal.

Table D-46

### KI UpDown Counter Config




















	<b>device:</b> Number of the device (beginning from 1). The utility <b>Device Browser</b> can be used to get the information of current device configuration.
	<b>sub type:</b> is the sub-type of the device you assigned to the Keithley Instruments PXI device during configuration.
	<b>counter:</b> is the counter this VI controls.
	<b>gate mode:</b> specifies how the signal on the counter's GATE pin is used. 0: Ungated/software start: ignore the gate source and start when Counter Start VI is called (default). 1: Count while the gate signal is TTL high after the KI Counter Start VI is called.
	<b>source edge:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
	<b>output:</b> This input is not used by Keithley Instruments PXI devices and is ignored.
	<b>updown source:</b> specifies how the signal on the counter's UPDN pin is used. 0: software control: ignore the UPDN source and control by updown control (default). 1: hardware control.
	<b>updown control:</b> specifies the specified counter to count down or count up if updown source is configured to be software control. 0: count down (default). 1: count up.
	<b>timebase:</b> (Hz) is set to the frequency of the internal signal whose cycles are counted, or is set to <=0.0 (default) to count the rising edges of the signal on the counter SOURCE pin.
	<b>error in (no error):</b> describes error conditions occurring before the VI executes. If an error has already occurred, the VI returns the value of the error in cluster in error out.
	<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
	<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
	<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.
	<b>count:</b> is the count value.
	<b>taskID:</b> identifies the group and the I/O operation.

Table D-46 (continued)  
**KI UpDown Counter Config**

	<b>error out:</b> contains error information. If the error in cluster indicated an error, the error out cluster contains the same information. Otherwise, error out describes the error status of this VI.
	<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
	<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
	<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.

## Advanced Counter VIs

### KI ICTR Control

This VI control counters on the Keithley Instruments PXI devices that use 82C54 chip. Control operations include starting, stopping, and setting the state of active acquisitions.

**NOTE** This VI is not supported for Keithley KDAQ series devices.

Table D-47  
**KI ICTR Control**





	<b>device:</b> Number of the device (beginning from 1). The utility <b>Device Browser</b> can be used to get the information of current device configuration.
	<b>sub type:</b> is the sub-type of the device you assigned to the Keithley Instruments PXI device during configuration.
	<b>counter:</b> is the counter this VI controls. KPXI-DIO-48: 0, 1 or 2
	control code: 0: Setup mode 0 – Toggle output from low to high on TC (default). 1: Setup mode 1 – Programmable one-shot. 2: Setup mode 2 – Rate generator. 3: Setup mode 3 – Square wave rate demerara. 4: Setup mode 4 – Software-triggered strobe. 5: Setup mode 5 – Hardware-triggered strobe. 6: Read. 7: Reset.

Table D-47 (continued)

**KI ICTR Control**

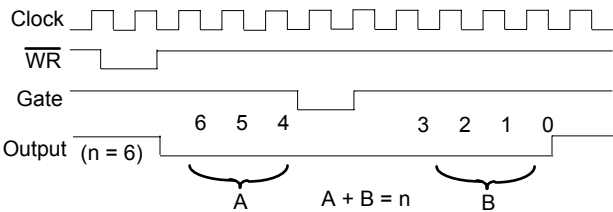
<b>U16</b>	<p><b>count:</b> is the period between output pulses. If control code is 0, 1, 4, or 5, count can be 0 through 65,535 in binary counter operation and 0 through 9,999 in binary-coded decimal (BCD) counter operation. If control code is 2 or 3, count can be 2 through 65,535 and 0 in binary counter operation and 2 through 9,999 and 0 in BCD counter operation.</p> <p><b>Setup mode 0: Toggle output from low to high on terminal count.</b> In this mode, as shown in the figure below, the output goes low after the mode set operation, and the counter begins to count down while the gate input is high. When terminal count is reached, the output goes high and remains high until the selected counter is set to a different mode.</p> 
------------	--

Table D-47 (continued)  
**KI ICTR Control**

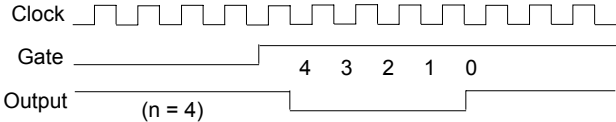
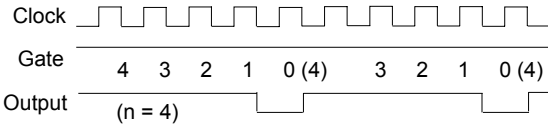
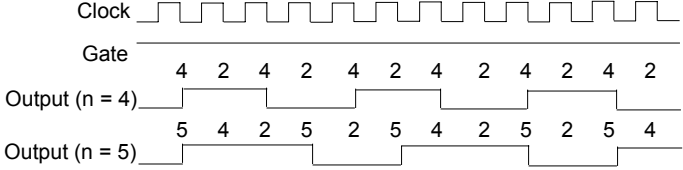
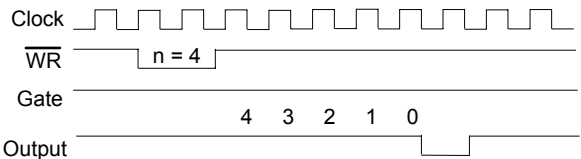
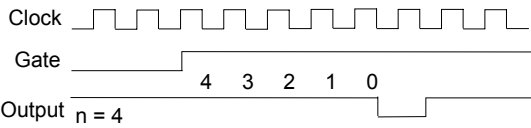














	<p><b>count:</b> (continued)</p> <p><b>Setup mode 1: Programmable one-shot</b>                  In this mode, as shown in the figure below, the <b>Output</b> goes low following the rising edge of <b>Gate</b> input <b>and</b> the falling edge of the clock. The <b>Output</b> and goes high on terminal count.</p>  <p><b>Setup mode 2: Rate generator</b>                  In this mode, the output goes low for one period of the clock input. <b>count</b> indicates the period from one output pulse to the next.</p>  <p><b>Setup mode 3: Square wave rate generator</b>                  In this mode, the output stays high for one half of the <b>count</b> clock pulses and stays low for the other half.</p>  <p><b>Setup mode 4: Software-triggered strobe</b>                  In this mode, the output is initially high, and the counter begins to count down while the gate input is high. On terminal count, the output goes low for one clock pulse, then goes high again. The following diagram shows the SOFT_TRIG mode timing diagram.</p>  <p><b>Setup mode 5: Hardware-triggered strobe</b>                  This mode is similar to Setup mode 4 except that the gate input is used as a trigger to start counting. The following diagram shows the HARD_TRIG mode timing diagram.</p> 
	<p><b>output state:</b> is only valid when control code is 7 (reset).                  0: Low (default input).                  1: High.</p>

Table D-47 (continued)

**KI ICTR Control**

	<b>binary or bcd:</b> controls whether the counter operates as a 16-bit binary counter or as a 4-decade BCD counter. 0: <i>BinBcd</i> . 1: 16-bit binary counter (default input)
	<b>Keithley Instruments PXI extensions:</b> additional features for Keithley Instruments PXI devices.
	<b>clock source:</b> defines the clock source for the timer/counter. 0: ECK1 (default input) 1: COUT n-1 2: CK1 3: COUT 10
	<b>error in (no error):</b> describes error conditions occurring before the VI executes. If an error has already occurred, the VI returns the value of the error in cluster in error out.
	<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
	<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
	<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.
	<b>read value:</b> When you set control code to 6 (read), read value returns the value the VI read from the counter.
	<b>taskID:</b> has the same value as <i>taskID in</i> .
	<b>error out:</b> contains error information. If the error in cluster indicated an error, the error out cluster contains the same information. Otherwise, error out describes the error status of this VI.
	<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
	<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
	<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.

# Calibration and Configuration VIs

## Calibration VIs

### KI KPXI-DAQ series devices and Digitizer Series Calibrate

Use this VI to calibrate KPXI-DAQ series devices and Digitizer Series device and to select a set of calibration constants to be used by KIDAQ LabVIEW.

**NOTE** This VI is not supported for Keithley KDIO devices.

Table D-48

### KI KPXI-DAQ series devices and Digitizer Series Calibrate














	<b>taskID in:</b> identifies the group and the I/O operation.
	<b>error in (no error):</b> describes error conditions occurring before the VI executes. If an error has already occurred, the VI returns the value of the error in cluster in error out.
	<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
	<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
	<b>Source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.
	<b>operation:</b> determines the operation the VI performs. 0: No change (default input). 1: Set default load area (default setting). 2: Self-calibrate. Setting the default load area, or value 1, does not perform a calibration; it sets the default load area to the area specified by calibration constants. Self-calibrate, or value 2, performs a calibration using the internal voltage reference.
	<b>calibration constants:</b> specifies which set of calibration constants KIDAQ LabVIEW uses. 0: No change (default input). 1: Factory EEPROM area, i.e. Bank 0 (default setting). 2: EEPROM Bank 0 area. 3: EEPROM Bank 1 area. 4: EEPROM Bank 2 area. 5: EEPROM Bank 3 area.
	<b>reference voltage:</b> this input is not used by Keithley Instruments PXI devices and is ignored.
	<b>taskID out:</b> has the same value as <i>taskID in</i> .
	<b>error out:</b> contains error information. If the error in cluster indicated an error, the error out cluster contains the same information. Otherwise, error out describes the error status of this VI.
	<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.

Table D-48 (continued)

**KI KPXI-DAQ series devices and Digitizer Series Calibrate**

	<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
	<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.

**Other Calibration and Configuration VIs****KI Route Signal**

Use this VI to route an internal signal to the specified I/O connector or SSI bus line, or to enable clock sharing through the SSI bus clock line.

**NOTE** This VI is not supported for Keithley KDIO devices.

Table D-49

**KI Route Signal**















	<b>taskID in:</b> identifies the group and the I/O operation.
	<b>error in (no error):</b> describes error conditions occurring before the VI executes. If an error has already occurred, the VI returns the value of the error in cluster in error out.
	<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
	<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
	<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.
	<p><b>signal name:</b> allows you to select the SSI line. The valid signal name for KPXI-DAQ series devices are as follows:</p> <ul style="list-style-type: none"> <li>0 : Do not change signal name (default input).</li> <li>1 : AI conversion.</li> <li>2 : AO update.</li> <li>3 : AI trigger.</li> <li>4 : AO trigger.</li> <li>19 : SSI Clock.</li> </ul> <p>The valid signal names for KPXI-AI-2-65M are the following:</p> <ul style="list-style-type: none"> <li>0 : Do not change signal name (default input).</li> <li>3 : AI trigger.</li> <li>19 : SSI Clock.</li> </ul>
	<b>signal name line number:</b> this input is not used by Keithley Instruments PXI devices and is ignored.



Table D-49 (continued)

**KI Route Signal**

	<p><b>signal source:</b> is the signal that KIDAQ LabVIEW routes to the location designated in signal name. There is only one valid signal source for most signal names. The valid signal source for KPXI-DAQ series devices are the following:                  0 : Do not change signal source (default input).                  1 : None (default setting).                  2 : AI Start Trigger.                  3 : AI Stop Trigger.                  4 : AI Convert.                  7 : AO Update.                  8 : AO Start Trigger.                  21 : Board Clock                  The valid signal sources for KPXI-AI-2-65M devices are the following:                  0 : Do not change signal source (default input).                  1 : None (default setting).                  2 : AI Start Trigger.                  3 : AI Stop Trigger.                  21 : Board Clock</p>
	<p><b>signal source line number:</b> this input is not used by Keithley Instruments PXI devices and is ignored.</p>
	<p><b>taskID out:</b> has the same value as <i>taskID in</i>.</p>
	<p><b>error out:</b> contains error information. If the error in cluster indicated an error, the error out cluster contains the same information. Otherwise, error out describes the error status of this VI.</p>
	<p><b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.</p>
	<p><b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.</p>
	<p><b>Source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.</p>

**KI SSI Control**

Connects or disconnects trigger and timing signals between DAQ devices along the Real-Time System Integration (SSI) bus.

**NOTE** *This VI is not supported for Keithley KDIO Series devices.*

Table D-50

**KI SSI Control**









	<p><b>taskID in:</b> identifies the group and the I/O operation.</p>
---	--

Table D-50 (continued)

**KI SSI Control**

	<p><b>board signal:</b> allows you to select the SSI line. The valid signal name for KPXI-DAQ series devices are as follows:</p> <ul style="list-style-type: none"> <li>0 : AI conversion.</li> <li>1 : AO update.</li> <li>2 : AI trigger.</li> <li>3 : AO trigger.</li> <li>4 : Board Clock.</li> <li>5 : AI Start</li> </ul> <p>The valid signal name for KPXI-AI-2-65M are the following:</p> <ul style="list-style-type: none"> <li>2 : AI trigger.</li> <li>4 : Board Clock.</li> </ul>
	<p><b>trigger line:</b> this input is not used by Keithley Instruments PXI devices and is ignored.</p>
	<p><b>direction:</b></p> <ul style="list-style-type: none"> <li>1 : The board transmits the signal to the bus.</li> </ul>
	<p><b>control code:</b></p> <ul style="list-style-type: none"> <li>0 : Do not change the control code setting (default input).</li> <li>1 : Clear.</li> <li>2 : Connect (default input).</li> <li>3 : Disconnect.</li> <li>4 : Construct the trigger line <b>usemap</b> only.</li> </ul>
	<p><b>device out:</b> has the same value as device.</p>
	<p><b>status:</b> This input is not used by KPXI-DAQ series devices and is ignored.</p>
	<p><b>trigger line usemap:</b> provides a list of free and busy SSI trigger lines. If trigger line <i>i</i> is not busy, <b>trigger line usemap[i]</b> shows a value of 0. If <b>trigger line <i>i</i></b> is busy, the VI sets <b>trigger line usemap[i]</b> to the device number of the device driving the line. Making only a receive connection to trigger line <i>i</i> does not set the [<i>i</i>]<b>th</b> element of trigger line <b>usemap</b>.</p>

## Service VIs

**KI Error Handler**

The KI Error Handler VI explains a non-zero error codes and shows dialog box with information about error. An error code equaling 0 (zero) means no error occurred.

Table D-51

**KI Error Handler**







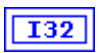

	<p><b>error in (no error):</b> describes error conditions occurring before the VI executes. If an error has already occurred, the VI returns the value of the error in cluster in error out.</p>
	<p><b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.</p>

Table D-51 (continued)  
**KI Error Handler**

	<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
	<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.
	<b>error out:</b> contains error information. If the error in cluster indicated an error, the error out cluster contains the same information. Otherwise, error out describes the error status of this VI.
	<b>status:</b> is TRUE if an error occurred. If status is TRUE, the VI does not perform any operations.
	<b>code:</b> is the error code number identifying an error. A value of 0 means no error, a negative value means a fatal error, and a positive value is a warning. Refer to <a href="#">Table D-52</a> for a code description.
	<b>source:</b> identifies where an error occurred. The source string is usually the name of the VI that produced the error.

## Error Codes

The Error Codes for KIDAQ LabVIEW VIs are contained in [Table D-52](#).

Table D-52  
 Error Codes: KIDAQ LabVIEW VIs

Code	Name	Description
0	NoError	No error occurred
-10006	badLineError	The line is invalid
-10007	badChanError	The value of AI/AO channel or DI/O port is invalid.
-10008	badGroupError	The group is invalid.
-10009	badCounterError	The value of input terminal <b>Ctr</b> is out of range.
-10010	badCountError	The value of input terminal <b>State</b> is out of range.
-10012	badRangeError	The specified A/D or D/A voltage value is out of range.
-10019	badClkFrequencyError	The frequency is invalid.
-10025	limitsOutOfRangeError	The value of AI range is invalid.
-10026	badBufferSpecificationError	The requested number of buffers or the buffer size is not allowed.
-10027	badDAQEventError	The DAQ events could not be opened.
-10041	badTaskIDError	The specified task ID is invalid.

Table D-52 (continued)  
Error Codes: KIDAQ LabVIEW VIs

Code	Name	Description
-10081	badPretrigCountError	The pretrigger sample count is invalid.
-10083	badTrigModeError	The trigger mode is invalid.
-10084	badTrigCountError	The trigger count is invalid.
-10086	badExtRefError	The external reference is invalid.
-10087	badTrigTypeError	The trigger type is invalid.
-10088	badTrigLevelError	The trigger level is invalid.
-10089	badTotalCountError	The DMA or interrupt transfer size is larger than the memory allocated in driver.
-10100	badPortWidthError	The requested digital port width is not a multiple of the hardware port width or is not attainable by the DAQ hardware.
-10121	gpctrBadCtrNumberError	Invalid <b>counterNumber</b> used.
-10122	gpctrBadParamValueError	Invalid <b>paramValue</b> used.
-10240	noDriverError	Open device driver failed.
-10242	functionNotFoundError	The function is not supported by this type of card.
-10341	badConnectError	The SSI signal/line cannot be connected as specified.
-10370	badScanListError	The scan list is invalid.
-10401	unknownDeviceError	The specified device is not a Keithley Instruments PXI device, the driver does not support the device.
-10402	deviceNotFoundError	No device is located in the specified slot or the device number is invalid.
-10409	groupBusyError	The specified group is in use.
-10411	counterBusyError	The specified counter is in use.
-10444	memFullError	Fail to allocate a driver internal use memory.
-10604	activeWriteError	Once data generation has started, only the transfer buffers originally written to may be updated.
-10608	noTransferInProgError	No transfer is in progress for the specified resource.
-10609	transferInProgError	A transfer is already in progress for the specified resource, or the operation is not allowed because the device is in the process of performing transfers, possibly with different resources.

Table D-52 (continued)  
 Error Codes: KIDAQ LabVIEW VIs

Code	Name	Description
-10612	badLineDirError	The specified line does not support the specified transfer direction.
-10613	badChanDirError	The specified channel does not support the specified transfer direction, or you have performed an operation on a digital port or line configured for the opposite direction.
-10618	badClkSrcError	The specified source signal cannot be assigned to the clock resource.
-10621	badTrigError	The specified trigger signal cannot be assigned to the trigger resource.
-10629	invalidOpModeError	The specified operating mode is invalid, or the resources have not been configured for the specified operating mode.
-10631	noInfiniteModeError	Continuous input or output transfers are not allowed in the current operating mode, or continuous operation is not allowed for this type of device.
-10634	noContTransferInProgressError	No continuous (double buffered) transfer is in progress.
-10636	noContWithSynchError	You cannot start a continuous (double-buffered) operation with a synchronous function call.
-10681	badChanRangeError	All channels of this board must have the same range.
-10697	rateNotSupportedError	The value of input terminal <b>SampleRate</b> is invalid.
-10800	timeOutError	The operation could not complete within the time limit.
-10801	calibrationError	An error occurred during the calibration process.
-10810	internalDriverError	An unexpected error occurred inside the driver when performing this given operation.
-10849	Unable to open a file	Fail to open a data file for storing input data.
-10856	osError	An unexpected error occurred from the operating system while performing the given operation.

## AI Range Codes

The Analog Input Range for Keithley Instruments PXI devices are contained in [Table D-53](#) and [Table D-54](#):

Table D-53  
**Analog Input Range**

Item	Range
1	Bipolar -10V to +10V

Table D-53 (continued)

**Analog Input Range**

Item	Range
2	Bipolar -5V to +5V
3	Bipolar -2.5V to +2.5V
4	Bipolar -1.25V to +1.25V
5	Bipolar -0.625V to +0.625V
6	Bipolar -0.3125V to +0.3125V
7	Bipolar -0.5V to +0.5V
8	Bipolar -0.05V to +0.05V
9	Bipolar -0.005V to +0.005V
10	Bipolar -1V to +1V
11	Bipolar -0.1V to +0.1V
12	Bipolar -0.01V to +0.01V
13	Bipolar -0.001V to +0.001V
14	Unipolar 0 to +20V
15	Unipolar 0 to +10V
16	Unipolar 0 to +5V
17	Unipolar 0 to +2.5V
18	Unipolar 0 to +1.25V
19	Unipolar 0 to +1V
20	Unipolar 0 to +0.1V
21	Unipolar 0 to +0.01V
22	Unipolar 0 to +0.001V
23	Bipolar -2V to +2V
24	Bipolar -0.25V to +0.25V
25	Bipolar -0.2V to +0.2V
26	Unipolar 0 to +4V
27	Unipolar 0 to +2V

Table D-53 (continued)

**Analog Input Range**

Item	Range
28	Unipolar 0 to +0.5V
29	Unipolar 0 to +0.4V

Table D-54

**Valid analog input ranges (specified by module)**

Model	Range
KPXI-SDAQ-4-500K KPXI-SDAQ-4-2M KPXI-DAQ-64-500K KPXI-DAQ-64-250K	1, 2, 3, 4, 15, 16, 17, 18
KPXI-DAQ-64-3M KPXI-DAQ-96-3M	1, 2, 3, 4, 7, 8, 10, 15, 16, 17, 19, 20, 23, 24, 25, 26, 27, 28, 29
KPXI-AO-4-1M KPXI-AO-8-1M	1, 15
KPXI-AI-2-65M	2, 10

## AI Data Format

Table D-55  
Analog Input data format (by Model)

Model	AI Data Format
KPXI-SDAQ-4-2M	<p>16-bit signed integer data: D13 D12 D11 ..... D1 D0 b1 b0</p> <p>Where: D13, D12, ... , D0 : A/D converted data b1, b0 : Simultaneous Digital Input data.</p>
KPXI-SDAQ-4-500K	<p>16-bit unsigned integer data: D15 D14 D13 ..... D1 D0</p> <p>Where: D15, D14, ... , D0 : A/D converted data</p>
KPXI-DAQ-64-3M	<p>16-bit signed integer data: D11 D10 D9 ..... D1 D0 b3 b2 b1 b0</p> <p>Where: D11, D10, ... , D0 : A/D converted data b3, b2, b1, b0 : Simultaneous Digital Input data.</p>
KPXI-DAQ-96-3M	<p>16-bit signed integer data: D11 D10 D9 ..... D1 D0 b3 b2 b1 b0</p> <p>Where: D11, D10, ... , D0 : A/D converted data b3, b2, b1, b0 : not used.</p>
KPXI-DAQ-64-500K KPXI-DAQ-64-250K	<p>16-bit signed integer data: D15 D14 D13 ..... D1 D0</p> <p>Where: D15, D14, ... , D0 : A/D converted data</p>
KPXI-AO-4-1M KPXI-AO-8-1M	<p>16-bit signed integer data: D13 D12 D11 ..... D1 D0 b1 b0</p> <p>Where D13, D12, ... , D0 : A/D converted data b1, b0 : AI Auto-scan Channel.</p>
KPXI-AI-2-65M	<p>16-bit unsigned integer data: b15 b14 D13 D12 D11 ..... D1D0</p> <p>Where: D13, D12, ... , D0 : A/D converted data b14 : Over-voltage indicator</p>



**Model No.** \_\_\_\_\_ **Serial No.** \_\_\_\_\_ **Date** \_\_\_\_\_

**Name and Telephone No.** \_\_\_\_\_

**Company** \_\_\_\_\_

List all control settings, describe problem and check boxes that apply to problem. \_\_\_\_\_

\_\_\_\_\_

Intermittent                       Analog output follows display                       Particular range or function bad; specify

IEEE failure                       Obvious problem on power-up                       Batteries and fuses are OK

Front panel operational    All ranges or functions are bad                       Checked all cables

Display or output (check one)

Drifts                       Unable to zero                       Unstable

Overload                       Will not read applied input

Calibration only                       Certificate of calibration required                       Data required

(attach any additional sheets as necessary)

Show a block diagram of your measurement including all instruments connected (whether power is turned on or not).  
Also, describe signal source.

Where is the measurement being performed? (factory, controlled laboratory, out-of-doors, etc.) \_\_\_\_\_

What power line voltage is used? \_\_\_\_\_ Ambient temperature? \_\_\_\_\_ °F

Relative humidity? \_\_\_\_\_ Other? \_\_\_\_\_

Any additional information. (If special modifications have been made by the user, please describe.)

\_\_\_\_\_

**Be sure to include your name and telephone number on this service form.**



Specifications are subject to change without notice.  
All Keithley trademarks and trade names are the property of Keithley Instruments, Inc.  
All other trademarks and trade names are the property of their respective companies.



A G R E A T E R M E A S U R E O F C O N F I D E N C E

**Keithley Instruments, Inc.**

**Corporate Headquarters** • 28775 Aurora Road • Cleveland, Ohio 44139 • 440-248-0400 • Fax: 440-248-6168 • 1-888-KEITHLEY • [www.keithley.com](http://www.keithley.com)