DAQ Analog Output Series

NI 6738/6739 User Manual

December 2017
375140B-01

NATIONAL INSTRUMENTS™

Worldwide Technical Support and Product Information
ni.com

Worldwide Offices
Visit ni.com/niglobal to access the branch office websites, which provide up-to-date contact information, support phone numbers, email addresses, and current events.

National Instruments Corporate Headquarters
11500 North Mopac Expressway Austin, Texas 78759-3504 USA Tel: 512 683 0100

For further support information, refer to the *Services and Resources* appendix. To comment on National Instruments documentation, refer to the National Instruments website at ni.com/info and enter the Info Code feedback.

Legal Information

Limited Warranty

This document is provided ‘as is’ and is subject to being changed, without notice, in future editions. For the latest version, refer to `ni.com/manuals`. NI reviews this document carefully for technical accuracy; however, NI MAKES NO EXPRESS OR IMPLIED WARRANTIES AS TO THE ACCURACY OF THE INFORMATION CONTAINED HEREIN AND SHALL NOT BE LIABLE FOR ANY ERRORS.

NI warrants that its hardware products will be free of defects in materials and workmanship that cause the product to fail to substantially conform to the applicable NI published specifications for one (1) year from the date of invoice.

For a period of ninety (90) days from the date of invoice, NI warrants that (i) its software products will perform substantially in accordance with the applicable documentation provided with the software and (ii) the software media will be free from defects in materials and workmanship.

If NI receives notice of a defect or non-conformance during the applicable warranty period, NI will, in its discretion: (i) repair or replace the affected product, or (ii) refund the fees paid for the affected product. Repaired or replaced Hardware will be warranted for the remainder of the original warranty period or ninety (90) days, whichever is longer. If NI elects to repair or replace the product, NI may use new or refurbished parts or products that are equivalent to new in performance and reliability and are at least functionally equivalent to the original part or product.

You must obtain an RMA number from NI before returning any product to NI. NI reserves the right to charge a fee for examining and testing Hardware not covered by the Limited Warranty.

This Limited Warranty does not apply if the defect of the product resulted from improper or inadequate maintenance, installation, repair, or calibration (performed by a party other than NI); unauthorized modification; improper environment; use of an improper hardware or software key; improper use or operation outside of the specification for the product; improper voltages; accident, abuse, or neglect; or a hazard such as lightning, flood, or other act of nature.

THE REMEDIES SET FORTH ABOVE ARE EXCLUSIVE AND THE CUSTOMER’S SOLE REMEDIES, AND SHALL APPLY EVEN IF SUCH REMEDIES FAIL OF THEIR ESSENTIAL PURPOSE.

EXCEPT AS EXPRESSLY SET FORTH HEREIN, PRODUCTS ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND AND NI DISCLAIMS ALL WARRANTIES, EXPRESSED OR IMPLIED, WITH RESPECT TO THE PRODUCTS, INCLUDING ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE OR NON-INFRINGEMENT, AND ANY WARRANTIES THAT MAY ARISE FROM USAGE OF TRADE OR COURSE OF DEALING. NI DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS REGARDING THE USE OF OR THE RESULTS OF THE USE OF THE PRODUCTS IN TERMS OF CORRECTNESS, ACCURACY, RELIABILITY, OR OTHERWISE. NI DOES NOT WARRANT THAT THE OPERATION OF THE PRODUCTS WILL BE UNINTERRUPTED OR ERROR FREE.

In the event that you and NI have a separate signed written agreement with warranty terms covering the products, then the warranty terms in the separate agreement shall control.

Copyright

End-User License Agreements and Third-Party Legal Notices

You can find end-user license agreements (EULAs) and third-party legal notices in the following locations:

- Notices are located in the `<National Instruments>\_Legal Information` and `<National Instruments>` directories.
- EULAs are located in the `<National Instruments>\Shared\MDF\Legal\license` directory.
- Review `<National Instruments>\_Legal Information.txt` for information on including legal information in installers built with NI products.

U.S. Government Restricted Rights

If you are an agency, department, or other entity of the United States Government (“Government”), the use, duplication, reproduction, release, modification, disclosure or transfer of the technical data included in this manual is governed by the Restricted Rights provisions under Federal Acquisition Regulation 52.227-14 for civilian agencies and Defense Federal Acquisition Regulation Supplement Section 252.227-7014 and 252.227-7015 for military agencies.

Trademarks

Refer to the *NI Trademarks and Logo Guidelines* at `ni.com/trademarks` for more information on National Instruments trademarks.

ARM, Keil, and µVision are trademarks or registered of ARM Ltd or its subsidiaries.

LEGO, the LEGO logo, WEDO, and MINDSTORMS are trademarks of the LEGO Group.

TETRIX by Pitsco is a trademark of Pitsco, Inc.

FIELDBUS FOUNDATION™ and FOUNDATION™ are trademarks of the Fieldbus Foundation.

Patents

For patents covering National Instruments products/technology, refer to the appropriate location: **Help»Patents** in your software, the patents.txt file on your media, or the *National Instruments Patent Notice* at ni.com/patents.

Export Compliance Information

Refer to the *Export Compliance Information* at ni.com/legal/export-compliance for the National Instruments global trade compliance policy and how to obtain relevant HTS codes, ECCNs, and other import/export data.

WARNING REGARDING USE OF NATIONAL INSTRUMENTS PRODUCTS

YOU ARE ULTIMATELY RESPONSIBLE FOR VERIFYING AND VALIDATING THE SUITABILITY AND RELIABILITY OF THE PRODUCTS WHENEVER THE PRODUCTS ARE INCORPORATED IN YOUR SYSTEM OR APPLICATION, INCLUDING THE APPROPRIATE DESIGN, PROCESS, AND SAFETY LEVEL OF SUCH SYSTEM OR APPLICATION.

PRODUCTS ARE NOT DESIGNED, MANUFACTURED, OR TESTED FOR USE IN LIFE OR SAFETY CRITICAL SYSTEMS, HAZARDOUS ENVIRONMENTS OR ANY OTHER ENVIRONMENTS REQUIRING FAIL-SAFE PERFORMANCE, INCLUDING IN THE OPERATION OF NUCLEAR FACILITIES; AIRCRAFT NAVIGATION; AIR TRAFFIC CONTROL SYSTEMS; LIFE SAVING OR LIFE SUSTAINING SYSTEMS OR SUCH OTHER MEDICAL DEVICES; OR ANY OTHER APPLICATION IN WHICH THE FAILURE OF THE PRODUCT OR SERVICE COULD LEAD TO DEATH, PERSONAL INJURY, SEVERE PROPERTY DAMAGE OR ENVIRONMENTAL HARM (COLLECTIVELY, “HIGH-RISK USES”). FURTHER, PRUDENT STEPS MUST BE TAKEN TO PROTECT AGAINST FAILURES, INCLUDING PROVIDING BACK-UP AND SHUT-DOWN MECHANISMS. NI EXPRESSLY DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY OF FITNESS OF THE PRODUCTS OR SERVICES FOR HIGH-RISK USES.

Compliance

Electromagnetic Compatibility Information

This product was tested and complies with the regulatory requirements and limits for electromagnetic compatibility (EMC) stated in the product specifications. These requirements and limits provide reasonable protection against harmful interference when the product is operated in the intended operational electromagnetic environment.

This product is intended for use in industrial locations. However, harmful interference may occur in some installations, when the product is connected to a peripheral device or test object, or if the product is used in residential or commercial areas. To minimize interference with radio and television reception and prevent unacceptable performance degradation, install and use this product in strict accordance with the instructions in the product documentation.

Furthermore, any modifications to the product not expressly approved by National Instruments could void your authority to operate it under your local regulatory rules.

Notices To ensure the specified EMC performance, operate this product only with shielded cables and accessories.

To ensure the specified EMC performance, the length of all I/O cables must be no longer than 30 m (100 ft).

Contents

Chapter 1
Getting Started

Chapter 2
DAQ System Overview

Chapter 3
Connector Information

Chapter 4
Analog Output

Chapter 5
Digital I/O

Chapter 6
Counters

Chapter 7
PFI

Chapter 8
Digital Routing and Clock Generation

Chapter 9
Bus Interface

Appendix A
Where to Go from Here

Appendix B
NI 6738/6739 in an NI 6723 System (PXI Express Only)

Appendix C
NI Services

1 Getting Started

The *NI 6738/6739 User Manual* contains information about using the National Instruments data acquisition (DAQ) devices with NI-DAQmx. The NI 6738/6739 features up to 64 analog output (AO) channels, up to 20 lines of digital input/output (DIO), and four counters. This chapter provides basic information you need to get started using your device.

Notice This icon denotes a notice advising you to take precautions to avoid data loss, loss of signal integrity, or void of guaranteed specifications.

Caution This icon denotes a caution advising you take precautions to avoid injury.

Safety Guidelines

Operate the NI 6738/6739 devices and modules only as described in this user manual.

Caution NI 6738/6739 devices and modules are *not* certified for use in hazardous locations.

Caution *Never* connect the +5 V power terminals to analog or digital ground or to any other voltage source on the NI 6738/6739 device or any other device. Doing so can damage the device and the computer. NI is not liable for damage resulting from such a connection.

Caution Exceeding the maximum input voltage ratings, which are listed in the specifications document for each NI 6738/6739 device, can damage the DAQ device and the computer. NI is not liable for any damage resulting from such signal connections.

Caution Damage can result if these lines are driven by the sub-bus. NI is not liable for any damage resulting from improper signal connections.

Safety Guidelines for Hazardous Voltages

If *hazardous voltages* are connected to the device/module, take the following precautions. A hazardous voltage is a voltage greater than 42.4 V_{pk} or 60 VDC to earth ground.

Caution Ensure that hazardous voltage wiring is performed only by qualified personnel adhering to local electrical standards.

Caution Do *not* mix hazardous voltage circuits and human-accessible circuits on the same module.

Caution Make sure that chassis and circuits connected to the module are properly insulated from human contact.

Caution NI 6738/6739 devices and modules provide no isolation.

Electromagnetic Compatibility Information

This product was tested and complies with the regulatory requirements and limits for electromagnetic compatibility (EMC) stated in the product specifications. These requirements and limits provide reasonable protection against harmful interference when the product is operated in the intended operational electromagnetic environment.

This product is intended for use in industrial locations. However, harmful interference may occur in some installations, when the product is connected to a peripheral device or test object, or if the product is used in residential or commercial areas. To minimize interference with radio and television reception and prevent unacceptable performance degradation, install and use this product in strict accordance with the instructions in the product documentation.

Furthermore, any modifications to the product not expressly approved by National Instruments could void your authority to operate it under your local regulatory rules.

Notices To ensure the specified EMC performance, operate this product only with shielded cables and accessories.

To ensure the specified EMC performance, the length of all I/O cables must be no longer than 30 m (100 ft).

Hardware Symbol Definitions

The following symbols are marked on your device or module.

Caution When this symbol is marked on a product, refer to the *Safety Guidelines* section for information about precautions to take.

EU Customers At the end of the product life cycle, all products *must* be sent to a WEEE recycling center. For more information about WEEE recycling centers, National Instruments WEEE initiatives, and compliance with WEEE Directive

2002/96/EC on Waste and Electronic Equipment, visit ni.com/environment/weee.

中国客户 National Instruments 符合中国电子信息产品中限制使用某些有害物质指令(RoHS)。关于 National Instruments 中国 RoHS 合规性信息，请登录 ni.com/environment/rohs_china。 (For information about China RoHS compliance, go to ni.com/environment/rohs_china.)

Installation

Before installing your DAQ device, you must install the software you plan to use with the device.

1. **Installing application software**—Refer to the installation instructions that accompany your software.
2. **Installing NI-DAQmx**—The *DAQ Getting Started* guides, packaged with NI-DAQmx and also on ni.com/manuals, contain step-by-step instructions for installing software and hardware, configuring channels and tasks, and getting started developing an application.
3. **Installing the hardware**—Unpack your device as described in the *Unpacking* section. The *DAQ Getting Started* guides describe how to install the device, as well as accessories and cables.

Unpacking

The NI 6738/6739 device ships in an antistatic package to prevent electrostatic discharge (ESD). ESD can damage several components on the device.

Caution *Never* touch the exposed pins of connectors.

To avoid ESD damage in handling the device, take the following precautions:

- Ground yourself with a grounding strap or by touching a grounded object.
- Touch the antistatic package to a metal part of your computer chassis before removing the device from the package.

Remove the device from the package and inspect it for loose components or any other signs of damage. Notify NI if the device appears damaged in any way. Do not install a damaged device in your computer or chassis.

Store the device in the antistatic package when the device is not in use.

Device Self-Calibration

NI recommends that you self-calibrate your device after installation and whenever the ambient temperature changes. Self-calibration should be performed after the device has warmed up for the recommended time period. Refer to the device specifications to find your device warm-up time. This function measures the onboard reference voltage of the device and adjusts the

self-calibration constants to account for any errors caused by short-term fluctuations in the environment. Disconnect all external signals when you self-calibrate a device.

You can initiate self-calibration using NI Measurement & Automation Explorer (MAX), by completing the following steps.

1. Launch MAX.
2. Select **My System»Devices and Interfaces»*your device***.
3. Initiate self-calibration using one of the following methods:
   - Click **Self-Calibrate** in the upper right corner of MAX.
   - Right-click the name of the device in the MAX configuration tree and select **Self-Calibrate** from the drop-down menu.

Note You can also programmatically self-calibrate your device with NI-DAQmx, as described in *Device Calibration* in the *NI-DAQmx Help* or the *LabVIEW Help*.

Device Pinouts

Refer to Chapter 3, *Connector Information*, for device pinouts.

Device Specifications

Refer to the device specifications document for your device. To locate your NI 6738/6739 device documentation, go to `ni.com/manuals` and search for your device:

- PCIe-6738
- PXIe-6738
- PXIe-6739

Device Accessories and Cables

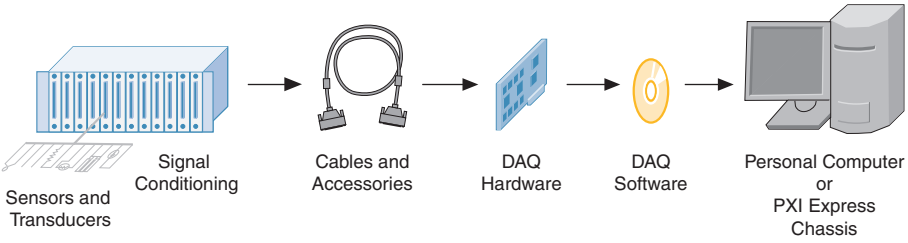
NI offers a variety of accessories and cables to use with your DAQ device. Refer to the *Cables and Accessories* section of Chapter 2, *DAQ System Overview*, for more information.

2

DAQ System Overview

Figure 2-1 shows a typical DAQ system, which includes sensors, transducers, signal conditioning devices, cables that connect the various devices to the accessories, the NI 6738/6739 device, programming software, and PC. The following sections cover the components of a typical DAQ system.

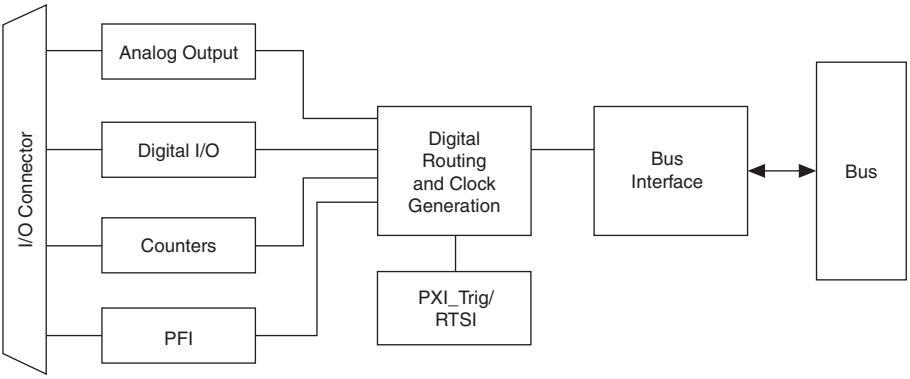
Figure 2-1. Components of a Typical DAQ System



DAQ Hardware

DAQ hardware digitizes signals, performs D/A conversions to generate analog output signals, and measures and controls digital I/O signals. Figure 2-2 features components common to most DAQ devices.

Figure 2-2. General DAQ Device Block Diagram



DAQ-STC3

The DAQ-STC3 used in the NI 6738/6739 implements a high-performance digital engine for DAQ data acquisition hardware. Some key features of this engine include the following:

- Flexible AO sample and convert timing
- Many triggering modes
- Independent AO, DI, DO, and counter FIFOs
- Generation and routing of RTSI or PXI_Trig signals for multi-device synchronization
- Generation and routing of internal and external timing signals
- Four flexible 32-bit counter/timer modules with hardware gating
- Digital waveform acquisition and generation
- Static DIO signals
- True 5 V high current drive DO
- DI change detection
- DO watchdog timers
- PLL for clock synchronization
- Seamless interface to signal conditioning accessories
- PCI Express/PXI Express interface
- Independent scatter-gather DMA controllers for all acquisition and generation functions

Calibration Circuitry

The NI 6738/6739 analog outputs have calibration circuitry to correct gain and offset errors. You can calibrate the device to minimize AO errors caused by time and temperature drift at run time. No external circuitry is necessary; an internal reference ensures high accuracy and stability over time and temperature changes.

Factory-calibration constants are permanently stored in an onboard EEPROM and cannot be modified. When you self-calibrate the device, as described in the *Device Self-Calibration* section of Chapter 1, *Getting Started*, software stores new constants in a user-modifiable section of the EEPROM. To return a device to its initial factory calibration settings, software can copy the factory-calibration constants to the user-modifiable section of the EEPROM. Refer to the *NI-DAQmx Help* or the *LabVIEW Help* for more information about using calibration constants.

For a detailed calibration procedure for NI 6738/6739 devices, refer to the *NI 6738/6739 Calibration Procedure* available at `ni.com/manuals`.

Cables and Accessories

Caution For compliance with Electromagnetic Compatibility (EMC) requirements, this product must be operated with shielded cables and accessories. If unshielded cables or accessories are used, the EMC specifications are no longer guaranteed unless all unshielded cables and/or accessories are installed in a shielded enclosure with properly designed and shielded input/output ports.

NI offers a variety of products to use with the NI 6738/6739, including cables, connector blocks, and other accessories, as follows:

- Shielded cable assemblies
- Screw terminal connector blocks
- RTSI bus cable
- I/O connector adapters

For more specific information about these products, refer to `ni.com`.

Refer to the *Custom Cabling and Connectivity* section of this chapter for information about how to select accessories for your device.

PCI Express and PXI Express Device Cables and Accessories

This section describes some cable and accessory options for devices with one or more 68-pin connectors. Refer to `ni.com/info` and enter the Info Code `AOCables` for a complete list of all accessory options.

Screw Terminal Accessories

National Instruments offers several styles of screw terminal connector blocks. All terminal connector blocks require a cable to connect the NI 6738/6739 to a connector block, as listed in Table 2-1.

Table 2-1. Screw Terminal Accessories

Screw Terminal Accessory	Description
CB-68LP and CB-68LPR	Unshielded connector blocks
SCB-68A	Shielded connector block
TBX-68	DIN rail-mountable connector block

RTSI Cables

A RTSI bus cable connects timing and synchronization signals among PCI Express devices. Since PXI Express devices use PXI backplane signals for timing and synchronization, no cables are required.

Cables

You can use the following cables:

- SHC68-68-A2— Shielded 68-pin, with separate shielding around analog and digital cable sections.
- SH68-C68-S—Only for use integrating the NI 6738/6739 into a NI 6723 system. Refer to the *Connecting the NI 6738/6739 in a NI 6723 System* section for more information.

Custom Cabling and Connectivity

The CA-1000 is a configurable enclosure that gives user-defined connectivity and flexibility through customized panelettes. Visit `ni.com` for more information about the CA-1000.

NI offers cables and accessories for many applications. However, if you want to develop your own cable, adhere to the following guidelines for best results:

- Route the analog lines separately from the digital lines.
- To prevent noise when using a cable shield, use separate shields for the analog and digital sections of the cable.

For more information about the connectors used for DAQ devices, refer to the KnowledgeBase document, *Specifications and Manufacturers for Board Mating Connectors*, by going to `ni.com/info` and entering the Info Code `rdspmb`.

Connecting the NI 6738/6739 in a NI 6723 System

You can add the NI 6738/6739 to an existing system configured to use the NI 6723. For information on integrating the NI 6738/6739 into an existing NI 6723 system, refer to Appendix B, *NI 6738/6739 in an NI 6723 System (PXI Express Only)*.

Programming Devices in Software

National Instruments measurement devices are packaged with NI-DAQmx driver software, an extensive library of functions and VIs you can call from your application software, such as LabVIEW or LabWindows/CVI, to program all the features of your NI measurement devices. Driver software has an application programming interface (API), which is a library of VIs, functions, classes, attributes, and properties for creating applications for your device.

The NI 6738/6739 uses the NI-DAQmx driver. NI-DAQmx includes a collection of programming examples to help you get started developing an application. You can modify example code and save it in an application. You can use examples to develop a new application or add example code to an existing application.

To locate LabVIEW, LabWindows/CVI, Measurement Studio, Visual Basic, and ANSI C examples, refer to the document, *Where Can I Find NI-DAQmx Examples?*, by going to `ni.com/info` and entering the Info Code `daqmxexp`.

For additional examples, refer to `ni.com/examples`.

Table 2-2 lists the earliest NI-DAQmx support version for each device.

Table 2-2. X Series NI-DAQmx Software Support

Device	NI-DAQmx Earliest Version Support
NI PCIe-6738	NI-DAQmx 17.6
NI PXIe-6738/6739	NI-DAQmx 15.1

3

Connector Information

This chapter contains information on the NI 6738/6739 pinouts and information about the connector signals and power.

I/O Connector Pinouts

PCIe/PXIe-6738 Pinout

Figure 3-1 shows the pinout of the PCIe/PXIe-6738 device. For a detailed description of each signal, refer to the *I/O Connector Signal Descriptions* section.

Figure 3-1. PCIe/PXIe-6738 Pinout

CONNECTOR 0
(AO 0–31)

AO Bank	AO GND 30/31	68	34	AO 31
	AO 30	67	33	AO GND 28/29
	AO 29	66	32	AO 28
AO Bank	AO GND 26/27	65	31	AO 27
	AO 26	64	30	AO GND 24/25
	AO 25	63	29	AO 24
AO Bank	AO GND 22/23	62	28	AO 23
	AO 22	61	27	AO GND 20/21
	AO 21	60	26	AO 20
AO Bank	AO GND 18/19	59	25	AO 19
	AO 18	58	24	AO GND 16/17
	AO 17	57	23	AO 16
AO Bank	AO GND[1]	56	22	AO 15
	AO GND 14/15	55	21	AO 14
	AO 13	54	20	AO GND 12/13
	AO 12	53	19	AO GND[1]
AO Bank	AO 11	52	18	AO GND 11
	AO 10	51	17	AO 9
	AO GND 8/9/10	50	16	AO 8
AO Bank	AO GND 6/7	49	15	AO 7
	AO 6	48	14	AO GND 4/5
	AO 5	47	13	AO 4
AO Bank	AO GND 2/3	46	12	AO 3
	AO 2	45	11	AO GND 0/1
	AO 1	44	10	AO 0
	D GND[1]	43	9	PFI 7/P1.7
	D GND PFI 6/7	42	8	PFI 6/P1.6
	D GND PFI 4/5	41	7	PFI 5/P1.5
	PFI 4/P1.4	40	6	PFI 3/P1.3
	D GND PFI 2/3	39	5	PFI 2/P1.2
	PFI 1/P1.1	38	4	PFI 0/P1.0
	D GND PFI 0/1	37	3	P0.1
	D GND P0.0/0.1	36	2	P0.0
	D GND[1]	35	1	+5 V

[1] No connect when using the SHC68-68-A2 cable.

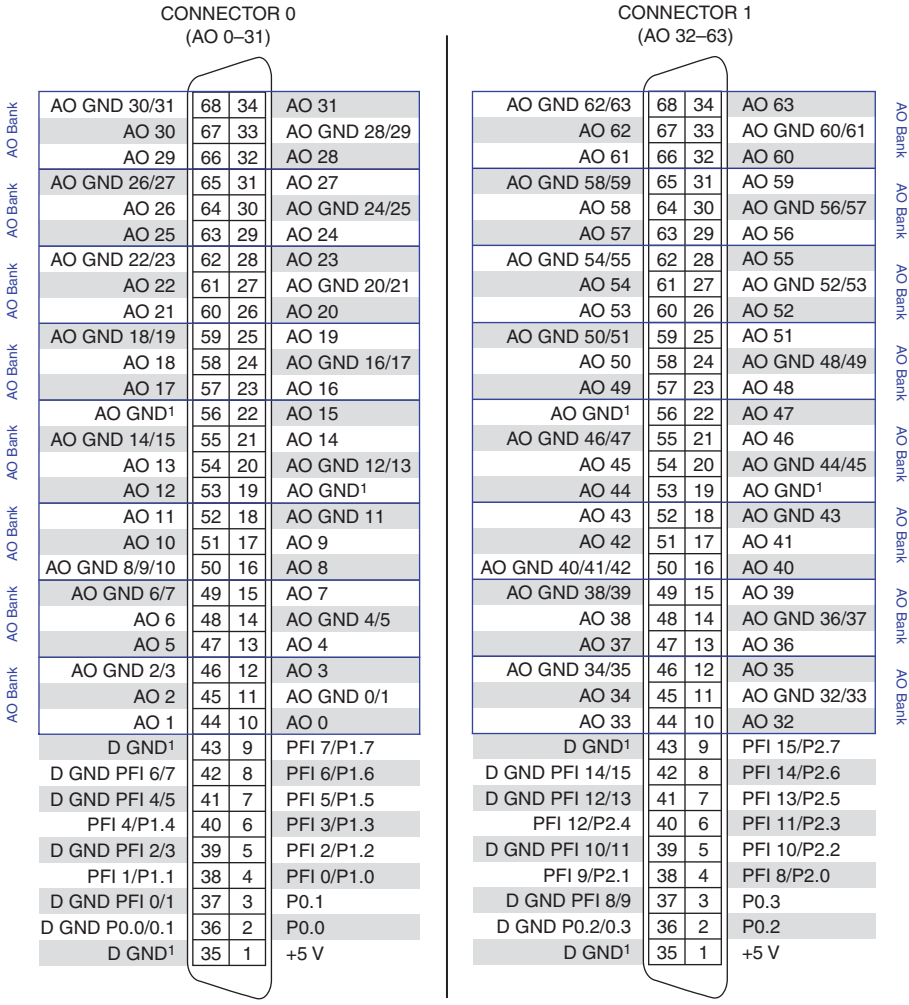
For the pin assignments of the PCIe/PXIe-6738 using the adapter and SH68-C68-S cable, refer to Appendix B, *NI 6738/6739 in an NI 6723 System (PXI Express Only)*.

Note For more information about default NI-DAQmx counter inputs, refer to *Connecting Counter Signals* in the *NI-DAQmx Help* or the *LabVIEW Help*.

PXIe-6739 Pinout

Figure 3-2 shows the pinout of the PXIe-6739. For a detailed description of each signal, refer to the *I/O Connector Signal Descriptions* section.

Figure 3-2. PXIe-6739 Pinout



[1] No connect when using the SHC68-68-A2 cable.

For the pin assignments of the PXIe-6739 using the adapter and SH68-C68-S cable, refer to Appendix B, *NI 6738/6739 in an NI 6723 System (PXI Express Only)*.

Note For more information about default NI-DAQmx counter inputs, refer to *Connecting Counter Signals* in the *NI-DAQmx Help* or the *LabVIEW Help*.

I/O Connector Signal Descriptions

Table 3-1 describes the signals found on the I/O connectors. Not all signals are available on all devices.

Table 3-1. I/O Connector Signals

Signal Name	Reference	Direction	Description
AO <0..63>	AO GND	Output	**Analog Output Lines 0 to 63**—These terminals supply the voltage output of AO lines 0 to 63.
AO GND	—	—	**Analog Output Ground**—AO GND is the reference for AO <0..63>. When AO GND is listed next to an analog signal name, it is the dedicated ground reference for those signals (e.g. AO GND 2/3 is the ground reference for AO 2 and AO 3).Both ground references—AO GND and D GND—are connected on the device.*
D GND	—	—	**Digital Ground**—D GND supplies the reference for P0.<0..3>, PFI <0..15>/P1/P2, and +5 V. When D GND is listed next to a digital signal name, it is the dedicated ground reference for those signals (e.g. D GND PFI 2/3 is the ground reference for PFI 2 and PFI 3). Both ground references—AO GND and D GND—are connected on the device.*
P0.<0..3>	D GND	Input or Output	**Port 0 Digital I/O Channels 0 to 3**—You can individually configure each signal as an input or output.
+5 V	D GND	Output	**+5 V Power Source**—These terminals provide a fused +5 V power source. Refer to the *+5 V Power Source* section for more information.
PFI <0..15>/ P1.<0..7> P2.<0..7>	D GND	Input or Output	**Programmable Function Interface or Digital I/O Lines 0 to 7 and Lines 8 to 15**—Each of these terminals can be individually configured as a PFI terminal or a digital I/O terminal. As an input, each PFI terminal can be used to supply an external source for AO, DI, and DO timing signals, or counter/timer inputs. As a PFI output, you can route many different internal AO, DI, or DO timing signals to each PFI terminal. You can also route the counter/timer outputs to each PFI terminal. As a Port 1 or Port 2 digital I/O signal, you can individually configure each signal as an input or output. Port 1 and Port 2 signals, however, do not support hardware-timed digital I/O.
NC	—	—	**No connect**—Do *not* connect signals to these terminals.

* Though AO GND and D GND are connected on the NI 6738/6739, they are connected by small traces to reduce crosstalk between subsystems. Each ground may have a slight difference in potential.

+5 V Power Source

The +5 V terminals on the I/O connector supply +5 V referenced to D GND. Use these terminals to power external circuitry.

Caution *Never* connect the +5 V power terminals to analog or digital ground or to any other voltage source on the NI 6738/6739 or any other device. Doing so can damage the device and the computer. NI is *not* liable for damage resulting from such a connection.

Refer to the specifications document for your device to obtain the device power rating.

RTSI Connector Pinout

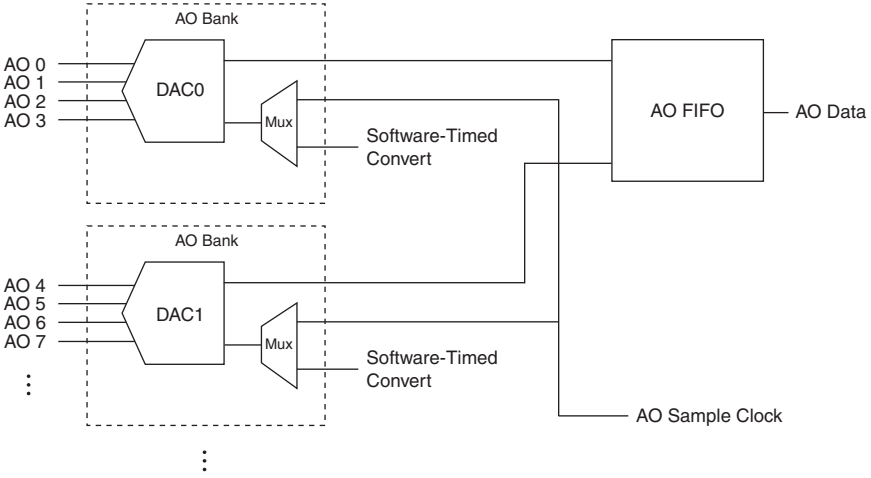
(PCIe-6738 Devices) Refer to the *RTSI Connector Pinout* section of Chapter 8, *Digital Routing and Clock Generation*, for information about the RTSI connector on the PCIe-6738 device.

4

Analog Output

NI 6738/6739 have either 32 or 64 AO channels that are controlled by a single clock and are capable of waveform generation. Figure 4-1 shows the analog output circuitry of the NI 6738/6739. Refer to the list below for detailed descriptions of the AO circuitry elements.

Figure 4-1. NI 6738/6739 Analog Output Circuitry



The main blocks featured in the NI 6738/6739 analog output circuitry are as follows:

- **DACs**—Digital-to-analog converters (DACs) convert digital data to analog voltages.
- **Banks**—Analog outputs are grouped into four-channel banks. Each bank of four channels is supported by one DAC per bank, and can select between the AO Sample Clock or a software-timed convert.
- **AO FIFO**—The AO FIFO enables analog output waveform generation. It is a first-in-first-out (FIFO) memory buffer between the computer and the DACs. It allows you to download the points of a waveform to your device without host computer interaction.
- **AO Sample Clock**—The AO Sample Clock is the convert source for hardware-timed tasks. Refer to the *Hardware-Timed Generations* section for more information.
- **Software-Timed Convert**—The software-timed convert causes updates on all of the banks in a software-timed task. Refer to the *Software-Timed Generations* section for more information.

Analog Output Data Generation Methods

When performing an analog output operation, each AO bank of four channels can operate in either software-timed or hardware-timed generations. Each bank can only perform one type of generation at a time.

Software-Timed Generations

With a software-timed generation, software controls the rate at which data is generated. Software sends a separate command to the hardware to initiate each DAC conversion. In NI-DAQmx, software-timed generations are referred to as on-demand timing. Software-timed generations are also referred to as immediate or static operations. They are typically used for writing a single value out, such as a constant DC voltage.

Software-timed tasks update all channels within their respective AO banks simultaneously. Each bank has a unique software-timed convert signal. A single on-demand task can update any combination of banks simultaneously. Multiple software-timed tasks can be run in parallel on separate banks.

Hardware-Timed Generations

With a hardware-timed generation, a digital signal controls the rate of the generation. This signal can be generated internally on your device or provided externally.

Note Only one hardware-timed generation can be performed on the NI 6738/6739 at a time.

Hardware-timed generations have several advantages over software-timed generations:

- The time between samples can be much shorter.
- The timing between samples can be deterministic.
- Hardware-timed generations can use hardware triggering.

Hardware-timed operations can be buffered or hardware-timed single point (HWTSP). A buffer is a temporary storage in computer memory for to-be-transferred samples.

- **Hardware-timed single point (HWTSP)**—HWTSP operations, used in conjunction with the wait for next sample clock function, provide tight synchronization between the software layer and the hardware layer. Typically, HWTSP operations are used to write single samples at known time intervals, which provides low latency and low jitter. In addition, HWTSP can notify software if it falls behind hardware in order to avoid writing stale samples. These features make HWTSP ideal for real time control applications such as hardware-in-the-loop (HIL). Refer to the *NI-DAQmx Hardware-Timed Single Point Lateness Checking* document for more information. To access this document, go to `ni.com/info` and enter the Info Code `daqhwtsp`.
- **Buffered**—In a buffered generation, data is moved from a PC buffer to the DAQ device's onboard FIFO using DMA. Buffered generation typically allow for much faster transfer

rates than non-buffered generations because data is moved in large blocks, rather than one point at a time.

One property of buffered I/O operations is the sample mode. The sample mode can be either finite or continuous:

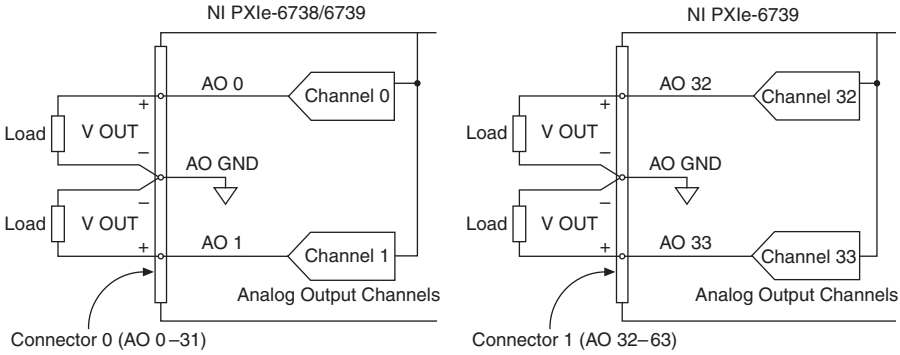
– Finite sample mode generations refers to generations of a specific, predetermined number of data samples. Once the specified number of samples has been written out, the generations stop.
– Continuous generations refers to generations of an unspecified number of samples. Instead of generating a set number of data samples and stopping, continuous generations continue until you stop the operation. There are several different methods of continuous generations that control what data is written. These methods are regeneration, FIFO regeneration and non-regeneration modes:
    • Regeneration is the repetition of the data that is already in the buffer. Standard regeneration is when data from the PC buffer is continually downloaded to the FIFO to be written out. New data can be written to the PC buffer at any time without disrupting the output. Use the NI-DAQmx write property RegenMode to allow (or not allow) regeneration. The NI-DAQmx default is to allow regeneration.
    • With FIFO regenerations, the entire buffer is downloaded to the FIFO and regenerated from there. Once the data is downloaded, new data cannot be written to the FIFO. To use FIFO regeneration, the entire buffer must fit within the FIFO size. The advantage of using FIFO regeneration is that it does not require communication with the main host memory once the operation is started, thereby preventing any problems that may occur due to excessive bus traffic. Use the NI-DAQmx AO channel property UseOnlyOnBoardMemory to enable or disable FIFO regeneration.
    • With non-regeneration, old data is not repeated. New data must be continually written to the buffer. If the program does not write new data to the buffer at a fast enough rate to keep up with the generations, the buffer underflows and causes an error.

Connecting Analog Output Signals

AO <0..31>/AO <0..63> are the voltage output signals for the analog output channels. AO GND is the ground reference for AO <0..31>/AO <0..63>.

Figure 4-2 shows how to make analog output connections to the device.

Figure 4-2. Analog Output Connections



Analog Output Triggering

Analog output supports two different triggering actions:

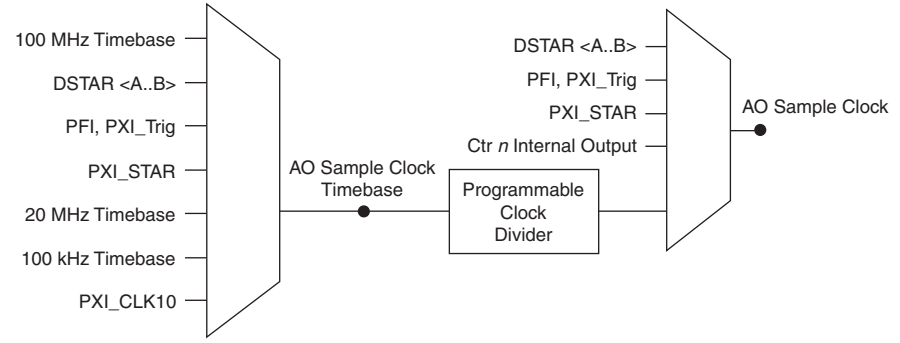
- Start trigger
- Pause trigger

A digital trigger can initiate these actions. Refer to the *AO Start Trigger Signal* and *AO Pause Trigger Signal* sections for more information about these triggering actions.

Analog Output Timing Signals

Figure 4-3 summarizes all of the timing options provided by the analog output timing engine.

Figure 4-3. Analog Output Timing Options



The NI 6738/6739 features the following analog output (waveform generation) timing signals:

- *AO Start Trigger Signal**
- *AO Pause Trigger Signal**

- *AO Sample Clock Signal**
- *AO Sample Clock Timebase Signal*

Signals with an * support digital filtering. Refer to the *PFI Filters* section of Chapter 7, *PFI*, for more information.

AO Start Trigger Signal

Use the AO Start Trigger (ao/StartTrigger) signal to initiate a waveform generation. If you do not use triggers, you can begin a generation with a software command.

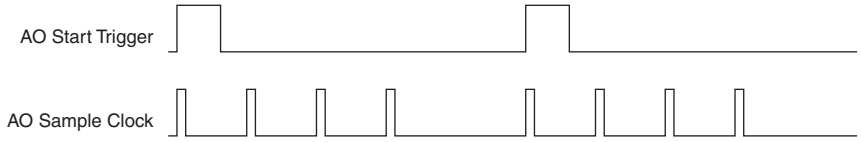
Retriggerable Analog Output

In Finite Mode, the AO Start Trigger is configurable as retriggerable. The timing engine generates the sample clock for the configured generation in response to each pulse on an AO Start Trigger signal.

The timing engine ignores the AO Start Trigger signal while the clock generation is in progress. After the clock generation is finished, the counter waits for another Start Trigger to begin another clock generation.

Figure 4-4 shows a retriggerable AO generation of four samples.

Figure 4-4. Retriggerable Analog Output



Using a Digital Source

To use AO Start Trigger, specify a source and an edge. The source can be one of the following signals:

- A pulse initiated by host software
- (NI PCIe/PXIe-6738) PFI <0..7>; (NI PXIe-6739) PFI <0..15>
- RTSI <0..7>
- PXI_Trig<0..7>
- PXI_STAR
- PXIe_DSTAR<A,B>
- Counter *n* Internal Output
- Change Detection Event
- DI Start Trigger (di/StartTrigger)
- DI Reference Trigger (di/ReferenceTrigger)
- DO Start Trigger (do/StartTrigger)

The source can also be one of several internal signals on your DAQ device. Refer to *Device Routing in MAX* in the *NI-DAQmx Help* or the *LabVIEW Help* for more information.

You can also specify whether the waveform generation begins on the rising edge or falling edge of AO Start Trigger.

Routing AO Start Trigger Signal to an Output Terminal

You can route AO Start Trigger out to any PFI <0..7>/PFI <0..15>, RTSI <0..7>, PXI_Trig<0..7>, or PXIe_DSTARC terminal.

The output is an active high pulse. PFI terminals are configured as inputs by default.

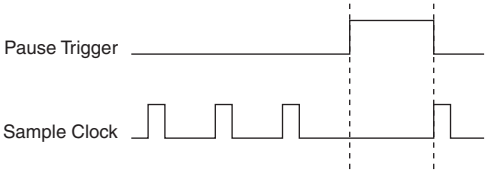
AO Pause Trigger Signal

Use the AO Pause Trigger (ao/PauseTrigger) signal to mask off sample clock pulses in a DAQ sequence. That is, when AO Pause Trigger is active, no updates occur.

AO Pause Trigger does not stop a sample that is in progress. The pause does not take effect until the beginning of the next sample.

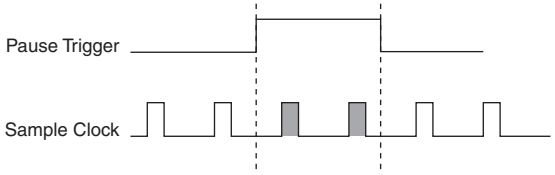
When you generate analog output signals, the generation pauses as soon as the pause trigger is asserted. If the source of your sample clock is the onboard clock, the generation resumes as soon as the pause trigger is deasserted, as shown in Figure 4-5. If you are performing a finite waveform output, the timing engine will continue counting samples during a pause trigger, even though no updates are occurring.

Figure 4-5. AO Pause Trigger with the Onboard Clock Source



If you are using any signal other than the onboard clock as the source of your sample clock, the generation resumes as soon as the pause trigger is deasserted and another edge of the sample clock is received, as shown in Figure 4-6.

Figure 4-6. AO PauseTrigger with Other Signal Source



Using a Digital Source

To use AO Pause Trigger, specify a source and a polarity. The source can be one of the following signals:

- (NI PCIe/PXIe-6738) PFI <0..7>; (NI PXIe-6739) PFI <0..15>
- RTSI <0..7>
- PXI_Trig<0..7>
- PXI_STAR
- PXIe_DSTAR<A,B>
- Counter *n* Internal Output
- Counter *n* Gate
- DI Pause Trigger (di/PauseTrigger)
- DO Pause Trigger (do/PauseTrigger)

The source can also be one of several other internal signals on your DAQ device. Refer to *Device Routing in MAX* in the *NI-DAQmx Help* or the *LabVIEW Help* for more information.

You can also specify whether the samples are paused when AO Pause Trigger is at a logic high or low level.

Routing AO Pause Trigger Signal to an Output Terminal

You can route AO Pause Trigger out to any PFI <0..7>/PFI <0..15>, RTSI <0..7>, PXI_Trig<0..7>, or PXIe_DSTARC terminal.

AO Sample Clock Signal

Use the AO Sample Clock (ao/SampleClock) signal to initiate AO samples. Each sample updates the outputs of all of the DACs on AO banks that are operating in hardware-timed generations. You can specify an internal or external source for AO Sample Clock. You can also specify whether the DAC update begins on the rising edge or falling edge of AO Sample Clock.

Using an Internal Source

One of the following internal signals can drive AO Sample Clock:

- AO Sample Clock Timebase (divided down)
- Counter *n* Internal Output
- Change Detection Event
- Counter *n* Sample Clock
- DI Sample Clock (di/SampleClock)
- DO Sample Clock (do/SampleClock)

A programmable internal counter divides down the AO Sample Clock Timebase signal.

Several other internal signals can be routed to AO Sample Clock through internal routes. Refer to *Device Routing in MAX* in the *NI-DAQmx Help* or the *LabVIEW Help* for more information.

Using an External Source

Use one of the following external signals as the source of AO Sample Clock:

- (NI PCIe/PXIe-6738) PFI <0..7>; (NI PXIe-6739) PFI <0..15>
- RTSI <0..7>
- PXI_Trig<0..7>
- PXI_STAR
- PXIe_DSTAR<A,B>

Routing AO Sample Clock Signal to an Output Terminal

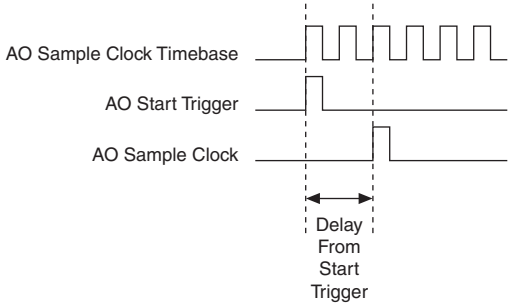
You can route AO Sample Clock (as an active low signal) out to any PFI <0..7>/PFI <0..15>, RTSI <0..7>, PXI_Trig<0..7>, or PXIe_DSTARC terminal.

Other Timing Requirements

The AO timing engine on your device internally generates AO Sample Clock unless you select some external source. AO Start Trigger starts the timing engine and either the software or hardware can stop it once a finite generation completes. When using the AO timing engine, you can also specify a configurable delay from AO Start Trigger to the first AO Sample Clock pulse. By default, this delay is two ticks of AO Sample Clock Timebase.

Figure 4-7 shows the relationship of AO Sample Clock to AO Start Trigger.

Figure 4-7. AO Sample Clock and AO Start Trigger



AO Sample Clock Timebase Signal

The AO Sample Clock Timebase (ao/SampleClockTimebase) signal is divided down to provide a source for AO Sample Clock.

You can route any of the following signals to be the AO Sample Clock Timebase signal:

- 100 MHz Timebase (default)
- 20 MHz Timebase
- 100 kHz Timebase
- PXI_CLK10
- (NI PCIe/PXIe-6738) PFI <0..7>; (NI PXIe-6739) PFI <0..15>
- RTSI <0..7>
- PXI_Trig<0..7>
- PXI_STAR
- PXIe_DSTAR<A,B>

AO Sample Clock Timebase is not available as an output on the I/O connector.

You might use AO Sample Clock Timebase if you want to use an external sample clock signal, but need to divide the signal down. If you want to use an external sample clock signal, but do not need to divide the signal, then you should use AO Sample Clock rather than AO Sample Clock Timebase.

Getting Started with AO Applications in Software

You can use the NI 6738/6739 in the following analog output applications:

- On-demand (single-point) generation
- Finite generation (waveform)
- Continuous generation (waveform)

You can perform these generations through programmed I/O or DMA data transfer mechanisms. Some of the applications also use start triggers and pause triggers.

Note For more information about programming analog output applications and triggers in software, refer to the *NI-DAQmx Help* or the *LabVIEW Help*.

The NI 6738/6739 uses the NI-DAQmx driver. NI-DAQmx includes a collection of programming examples to help you get started developing an application. You can modify example code and save it in an application. You can use examples to develop a new application or add example code to an existing application.

To locate LabVIEW, LabWindows/CVI, Measurement Studio, Visual Basic, and ANSI C examples, refer to the KnowledgeBase document, *Where Can I Find NI-DAQmx Examples?*, by going to `ni.com/info` and entering the Info Code `daqmxexp`.

For additional examples, refer to `ni.com/examples`.

5

Digital I/O

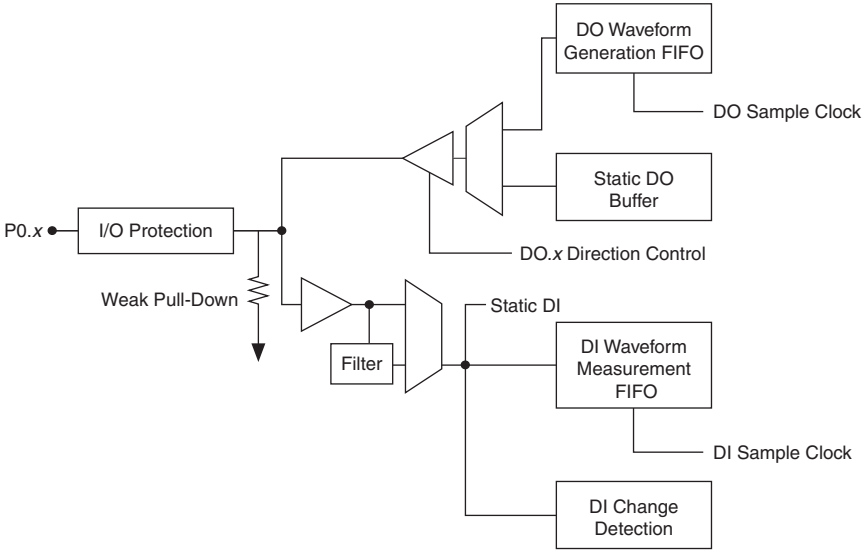
The NI 6738/6739 contains up to four lines of bidirectional DIO signals on Port 0. In addition, The NI 6738/6739 has up to 16 PFI signals that can function as static DIO signals.

The NI 6738/6739 supports the following DIO features on Port 0:

- Up to two lines of DIO on the NI PCIe/PXIe-6738, four lines of DIO on the NI PXIe-6739
- Direction and function of each terminal individually controllable
- Static digital input and output
- High-speed digital waveform generation
- High-speed digital waveform acquisition
- DI change detection trigger/interrupt

Figure 5-1 shows the circuitry of one DIO line. Each DIO line is similar. The following sections provide information about the various parts of the DIO circuit.

Figure 5-1. NI 6738/6739 Digital I/O Circuitry



The DIO terminals are named P0.<0..1> on the NI PCIe/PXIe-6738 I/O connector, and P0.<0..3> on the NI PXIe-6739 I/O connector.

The voltage input and output levels and the current drive levels of the DIO lines are listed in the specifications of your device.

Digital Input Data Acquisition Methods

When performing digital input measurements, you either can perform software-timed or hardware-timed acquisitions.

Software-Timed Acquisitions

With a software-timed acquisition, software controls the rate of the acquisition. Software sends a separate command to the hardware to initiate each acquisition. In NI-DAQmx, software-timed acquisitions are referred to as having on-demand timing. Software-timed acquisitions are also referred to as immediate or static acquisitions and are typically used for reading a single sample of data.

Each of the DIO lines can be used as a static DI or DO line. You can use static DIO lines to monitor or control digital signals. Each DIO can be individually configured as a digital input (DI) or digital output (DO).

All samples of static DI lines and updates of static DO lines are software-timed.

Hardware-Timed Acquisitions

With hardware-timed acquisitions, a digital hardware signal (di/SampleClock) controls the rate of the acquisition. This signal can be generated internally on your device or provided externally.

Hardware-timed acquisitions have several advantages over software-timed acquisitions.

- The time between samples can be much shorter.
- The timing between samples is deterministic.
- Hardware-timed acquisitions can use hardware triggering.

Hardware-timed operations can be buffered or hardware-timed single point. A buffer is a temporary storage in computer memory for to-be-transferred samples.

- **Buffered**—Data is moved from the DAQ device's onboard FIFO memory to a PC buffer using DMA before it is transferred to application memory. Buffered acquisitions typically allow for much faster transfer rates than non-buffered acquisitions because data is moved in large blocks, rather than one point at a time.

  One property of buffered I/O operations is the sample mode. The sample mode can be either finite or continuous:

  – Finite sample mode acquisition refers to the acquisition of a specific, predetermined number of data samples. Once the specified number of samples has been read in, the acquisition stops. If you use a reference trigger, you must use finite sample mode.

  – Continuous acquisition refers to the acquisition of an unspecified number of samples. Instead of acquiring a set number of data samples and stopping, a continuous

acquisition continues until you stop the operation. Continuous acquisition is also referred to as double-buffered or circular-buffered acquisition.

If data cannot be transferred across the bus fast enough, the FIFO becomes full. New acquisitions overwrites data in the FIFO before it can be transferred to host memory, which causes the device to generate an error. With continuous operations, if the user program does not read data out of the PC buffer fast enough to keep up with the data transfer, the buffer could reach an overflow condition, causing an error to be generated.

- **Hardware-timed single point (HWTSP)**—HWTSP operations, used in conjunction with the wait for next sample clock function, provide tight synchronization between the software layer and the hardware layer. Typically, HWTSP operations are used to read single samples at known time intervals, which provides low latency and low jitter. These features make HWTSP ideal for real time control applications such as hardware-in-the-loop (HIL). Refer to the *NI-DAQmx Hardware-Timed Single Point Lateness Checking* document for more information. To access this document, go to `ni.com/info` and enter the Info Code `daqhwtsp`.

Digital Input Triggering

Digital input supports three different triggering actions:

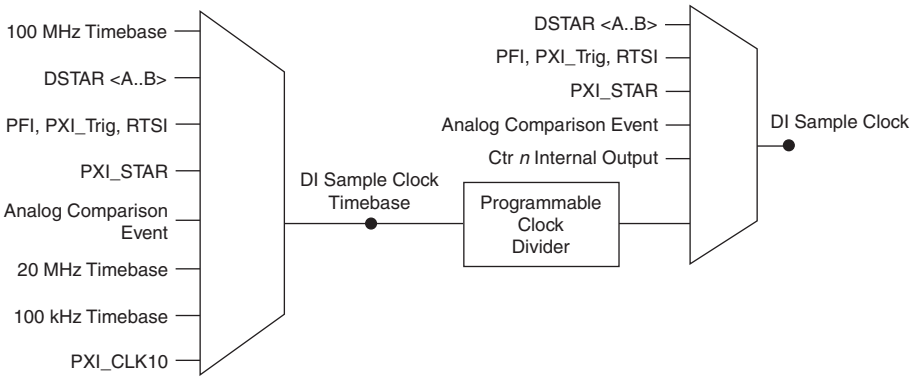
- Start trigger
- Reference trigger
- Pause trigger

Refer to the *DI Start Trigger Signal*, *DI Reference Trigger Signal*, and *DI Pause Trigger Signal* sections for information about these triggers.

Digital Waveform Acquisition

Figure 5-2 summarizes all of the timing options provided by the digital input timing engine.

Figure 5-2. Digital Input Timing Options



You can acquire digital waveforms on the Port 0 DIO lines. The DI waveform acquisition FIFO stores the digital samples. The NI 6738/6739 has a DMA controller dedicated to moving data from the DI waveform acquisition FIFO to system memory. The DAQ device samples the DIO lines on each rising or falling edge of a clock signal, DI Sample Clock.

You can configure each DIO line to be an output, a static input, or a digital waveform acquisition input.

The NI 6738/6739 features the following digital input timing signals:

- *DI Sample Clock Signal*[*]
- *DI Sample Clock Timebase Signal*
- *DI Start Trigger Signal*[*]
- *DI Reference Trigger Signal*[*]
- *DI Pause Trigger Signal*[*]

Signals with an [*] support digital filtering. Refer to the *PFI Filters* section of Chapter 7, *PFI*, for more information.

DI Sample Clock Signal

The device uses the DI Sample Clock (di/SampleClock) signal to sample the Port 0 terminals and store the result in the DI waveform acquisition FIFO.

You can specify an internal or external source for DI Sample Clock. You can also specify whether the measurement sample begins on the rising edge or falling edge of DI Sample Clock.

If the DAQ device receives a DI Sample Clock when the FIFO is full, it reports an overflow error to the host software.

Using an Internal Source

To use DI Sample Clock with an internal source, specify the signal source and the polarity of the signal. The source can be any of the following signals:

- DI Sample Clock (di/SampleClock)
- DO Sample Clock (do/SampleClock)
- Counter *n* Sample Clock
- Counter *n* Internal Output
- DI Change Detection output
- AO Sample Clock

Several other internal signals can be routed to DI Sample Clock through internal routes. Refer to *Device Routing in MAX* in the *NI-DAQmx Help* or the *LabVIEW Help* for more information.

Using an External Source

You can route any of the following signals as DI Sample Clock:

- (NI PCIe/PXIe-6738) PFI <0..7>; (NI PXIe-6739) PFI <0..15>
- RTSI <0..7>
- PXI_Trig<0..7>
- PXI_STAR
- PXIe_DSTAR<A,B>

You can sample data on the rising or falling edge of DI Sample Clock.

Routing DI Sample Clock to an Output Terminal

You can route DI Sample Clock out to any PFI <0..7>/<8..15> terminal. The PFI circuitry inverts the polarity of DI Sample Clock before driving the PFI terminal.

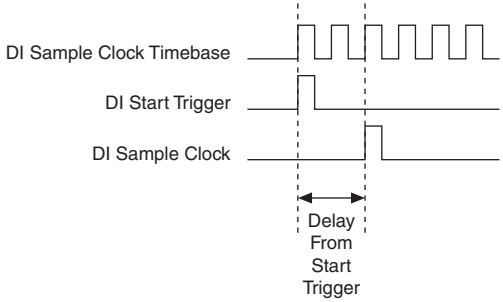
Other Timing Requirements

Your DAQ device only acquires data during an acquisition. The device ignores DI Sample Clock when a measurement acquisition is not in progress. During a measurement acquisition, you can cause your DAQ device to ignore DI Sample Clock using the DI Pause Trigger signal.

The DI timing engine on your device internally generates DI Sample Clock unless you select an external source. DI Start Trigger starts the timing engine and either software or hardware can stop it once a finite acquisition completes. When using the DI timing engine, you can also specify a configurable delay from DI Start Trigger to the first DI Sample Clock pulse.

By default, this delay is set to two ticks of the DI Sample Clock Timebase signal.

Figure 5-3. DI Sample Clock and DI Start Trigger



DI Sample Clock Timebase Signal

You can route any of the following signals to be the DI Sample Clock Timebase (di/SampleClockTimebase) signal:

- 100 MHz Timebase (default)
- 20 MHz Timebase
- 100 kHz Timebase
- PXI_CLK10
- RTSI <0..7>
- PXI_Trig<0..7>
- (NI PCIe/PXIe-6738) PFI <0..7>; (NI PXIe-6739) PFI <0..15>
- PXI_STAR
- PXIe_DSTAR<A,B>

Refer to the device routing table in MAX for all additional routable signals. To find the device routing table for your device, launch MAX and select **Devices and Interfaces»NI-DAQmx Devices**. Click a device to open a tabbed window in the middle pane. Click the **Device Routes** tab at the bottom of the pane to display the device routing table.

DI Sample Clock Timebase is not available as an output on the I/O connector. DI Sample Clock Timebase is divided down to provide one of the possible sources for DI Sample Clock. You can configure the polarity selection for DI Sample Clock Timebase as either rising or falling edge except for the 100 MHz Timebase or 20 MHz Timebase.

You might use DI Sample Clock Timebase if you want to use an external sample clock signal, but need to divide the signal down. If you want to use an external sample clock signal, but do not need to divide the signal, then you should use DI Sample Clock rather than DI Sample Clock Timebase.

DI Start Trigger Signal

Use the DI Start Trigger (di/StartTrigger) signal to begin a measurement acquisition. A measurement acquisition consists of one or more samples. If you do not use triggers, begin a measurement with a software command. Once the acquisition begins, configure the acquisition to stop:

- When a certain number of points are sampled (in finite mode)
- After a hardware reference trigger (in finite mode)
- With a software command (in continuous mode)

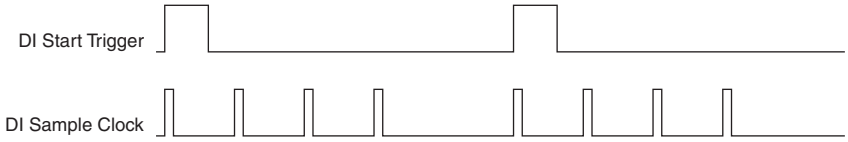
An acquisition that uses a start trigger (but not a reference trigger) is sometimes referred to as a posttriggered acquisition.

Retriggerable DI

When using finite sampling mode, the DI Start Trigger is configurable as retriggerable. When the DI Start Trigger is configured as retriggerable, the timing engine generates the sample clocks for the configured acquisition in response to each pulse on a DI Start Trigger signal.

The timing engine ignores the DI Start Trigger signal while the clock generation is in progress. After the clock generation is finished, the timing engine waits for another Start Trigger to begin another clock generation. Figure 5-4 shows a retriggerable DI of four samples.

Figure 5-4. Retriggerable DI



Note Waveform information from LabVIEW does not reflect the delay between triggers. They are treated as a continuous acquisition with constant t0 and dt information.

Reference triggers are not retriggerable.

Using a Digital Source

To use DI Start Trigger with a digital source, specify a source and an edge. The source can be any of the following signals:

- (NI PCIe/PXIe-6738) PFI <0..7>; (NI PXIe-6739) PFI <0..15>
- RTSI <0..7>
- PXI_Trig<0..7>
- Counter *n* Internal Output
- PXI_STAR

- PXIe_DSTAR<A,B>
- Change Detection Event
- DO Start Trigger (do/StartTrigger)
- AO Start Trigger

The source can also be one of several other internal signals on your DAQ device. Refer to *Device Routing in MAX* in the *NI-DAQmx Help* or the *LabVIEW Help* for more information.

You can also specify whether the measurement acquisition begins on the rising edge or falling edge of DI Start Trigger.

Routing DI Start Trigger to an Output Terminal

You can route DI Start Trigger out to any PFI <0..7>/<8..15>, RTSI <0..7>, PXI_Trig<0..7>, or PXIe_DSTARC terminal. The output is an active high pulse. All PFI terminals are configured as inputs by default.

The device also uses DI Start Trigger to initiate pretriggered DAQ operations. In most pretriggered applications, a software trigger generates DI Start Trigger. Refer to the *DI Reference Trigger Signal* section for a complete description of the use of DI Start Trigger and DI Reference Trigger in a pretriggered DAQ operation.

DI Reference Trigger Signal

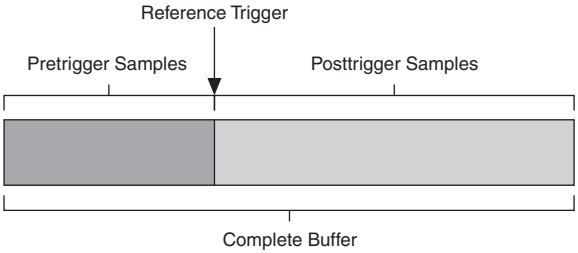
Use the DI Reference Trigger (di/ReferenceTrigger) signal to stop a measurement acquisition. To use a reference trigger, specify a buffer of finite size and a number of pretrigger samples (samples that occur before the reference trigger). The number of posttrigger samples (samples that occur after the reference trigger) desired is the buffer size minus the number of pretrigger samples.

Once the acquisition begins, the DAQ device writes samples to the buffer. After the DAQ device captures the specified number of pretrigger samples, the DAQ device begins to look for the reference trigger condition. If the reference trigger condition occurs before the DAQ device captures the specified number of pretrigger samples, the DAQ device ignores the condition.

If the buffer becomes full, the DAQ device continuously discards the oldest samples in the buffer to make space for the next sample. This data can be accessed (with some limitations) before the DAQ device discards it. Refer to the KnowledgeBase document, *Can a Pretriggered Acquisition be Continuous?*, for more information. To access this KnowledgeBase, go to `ni.com/info` and enter the Info Code `rdcanq`.

When the reference trigger occurs, the DAQ device continues to write samples to the buffer until the buffer contains the number of posttrigger samples desired. Figure 5-5 shows the final buffer.

Figure 5-5. Reference Trigger Final Buffer



Using a Digital Source

To use DI Reference Trigger with a digital source, specify a source and an edge. The source can be any of the following signals:

- (NI PCIe/PXIe-6738) PFI <0..7>; (NI PXIe-6739) PFI <0..15>
- RTSI <0..7>
- PXI_Trig<0..7>
- PXI_STAR
- PXIe_DSTAR<A,B>
- Change Detection Event
- Counter *n* Internal Output
- DO Start Trigger (do/StartTrigger)
- AO Start Trigger

The source can also be one of several internal signals on your DAQ device. Refer to *Device Routing in MAX* in the *NI-DAQmx Help* or the *LabVIEW Help* for more information.

You can also specify whether the measurement acquisition stops on the rising or falling edge or falling edge of DI Reference Trigger.

Routing DI Reference Trigger Signal to an Output Terminal

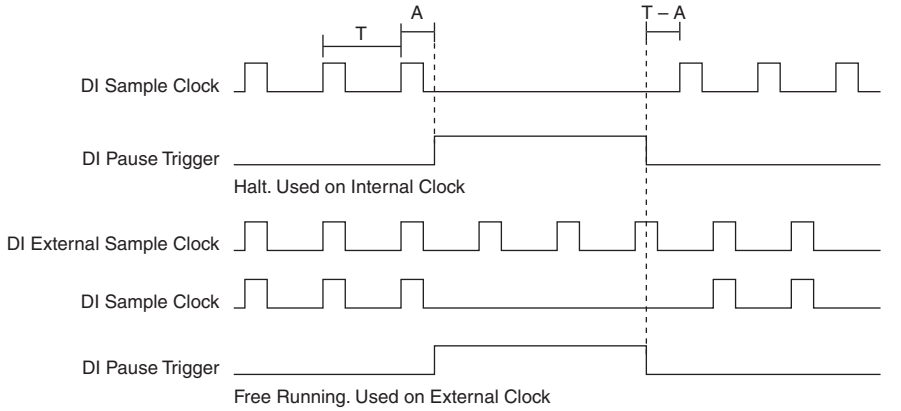
You can route DI Reference Trigger out to any PFI <0..7>/<8..15>, RTSI <0..7>, PXI_Trig<0..7>, or PXI_Trig <0..7>, PXIe_DSTARC terminal. All PFI terminals are configured as inputs by default.

DI Pause Trigger Signal

You can use the DI Pause Trigger (di/PauseTrigger) signal to pause and resume a measurement acquisition. The internal sample clock pauses while the external trigger signal is active and resumes when the signal is inactive. You can program the active level of the pause trigger to be high or low, as shown in Figure 5-6. In the figure, T represents the period, and A represents the unknown time between the clock pulse and the posttrigger. If you are performing a finite

waveform output, the timing engine will continue counting samples during a pause trigger, even though no updates are occurring.

Figure 5-6. Halt (Internal Clock) and Free Running (External Clock)



Using a Digital Source

To use DI Pause Trigger, specify a source and a polarity. The source can be any of the following signals:

- (NI PCIe/PXIe-6738) PFI <0..7>; (NI PXIe-6739) PFI <0..15>
- RTSI <0..7>
- PXI_Trig<0..7>
- PXI_STAR
- PXIe_DSTAR<A,B>
- Counter *n* Internal Output
- Counter *n* Gate
- DO Pause Trigger (do/PauseTrigger)
- AO Pause Trigger

The source can also be one of several other internal signals on your DAQ device. Refer to *Device Routing in MAX* in the *NI-DAQmx Help* or the *LabVIEW Help* for more information.

Routing DI Pause Trigger Signal to an Output Terminal

You can route DI Pause Trigger out to any PFI <0..7>/<8..15>, RTSI <0..7>, PXI_Trig<0..7>, PXI_STAR, or PXIe_DSTARC terminal.

Note Pause triggers are only sensitive to the level of the source, not the edge.

Digital Output Data Generation Methods

When performing a digital waveform operation, you either can perform software-timed or hardware-timed generations.

Software-Timed Generations

With a software-timed generation, software controls the rate at which data is generated. Software sends a separate command to the hardware to initiate each update. In NI-DAQmx, software-timed generations are referred to as on-demand timing. Software-timed generations are also referred to as immediate or static operations. They are typically used for writing a single value out, such as a constant digital value.

Hardware-Timed Generations

With a hardware-timed generation, a digital hardware signal controls the rate of the generation. This signal can be generated internally on your device or provided externally.

Hardware-timed generations have several advantages over software-timed generations:

- The time between samples can be much shorter.
- The timing between samples can be deterministic.
- Hardware-timed generations can use hardware triggering.

Hardware-timed operations can be buffered or hardware-timed single point (HWTSP). A buffer is a temporary storage in computer memory for to-be-transferred samples.

- **Hardware-timed single point (HWTSP)**—HWTSP operations, used in conjunction with the wait for next sample clock function, provide tight synchronization between the software layer and the hardware layer. Typically, HWTSP operations are used to write single samples at known time intervals, which provides low latency and low jitter. In addition, HWTSP can notify software if it falls behind hardware in order to avoid writing stale samples. These features make HWTSP ideal for real time control applications such as hardware-in-the-loop (HIL). Refer to the *NI-DAQmx Hardware-Timed Single Point Lateness Checking* document for more information. To access this document, go to `ni.com/info` and enter the Info Code `daqhwtsp`.
- **Buffered**—In a buffered generation, data is moved from a PC buffer to the DAQ device's onboard FIFO using DMA before it is written to the output lines one sample at a time. Buffered generation typically allow for much faster transfer rates than non-buffered acquisitions because data is moved in large blocks, rather than one point at a time.

  One property of buffered I/O operations is the sample mode. The sample mode can be either finite or continuous:

  – Finite sample mode generation refers to the generation of a specific, predetermined number of data samples. Once the specified number of samples has been written out, the generation stops.

– Continuous generation refers to the generation of an unspecified number of samples. Instead of generating a set number of data samples and stopping, a continuous generation continues until you stop the operation. There are several different methods of continuous generation that control what data is written. These methods are regeneration, FIFO regeneration and non-regeneration modes:
  - Regeneration is the repetition of the data that is already in the buffer. Standard regeneration is when data from the PC buffer is continually downloaded to the FIFO to be written out. New data can be written to the PC buffer at any time without disrupting the output. Use the NI-DAQmx write property regenMode to allow (or not allow) regeneration. The NI-DAQmx default is to allow regeneration.
  - With non-regeneration, old data is not repeated. New data must be continually written to the buffer. If the program does not write new data to the buffer at a fast enough rate to keep up with the generation, the buffer underflows and causes an error.
  - With FIFO regeneration, the entire buffer is downloaded to the FIFO and regenerated from there. Once the data is downloaded, new data cannot be written to the FIFO. To use FIFO regeneration, the entire buffer must fit within the FIFO size. The advantage of using FIFO regeneration is that it does not require communication with the main host memory once the operation is started, thereby preventing any problems that may occur due to excessive bus traffic. Use the NI-DAQmx UseOnlyOnBoardMemory DO channel property to enable or disable FIFO regeneration.

Digital Output Triggering

Digital output supports two different triggering actions:

- Start trigger
- Pause trigger

Digital Waveform Generation

You can generate digital waveforms on the Port 0 DIO lines. The DO waveform generation FIFO stores the digital samples. The NI 6738/6739 has a DMA controller dedicated to moving data from the system memory to the DO waveform generation FIFO. The DAQ device moves samples from the FIFO to the DIO terminals on each rising or falling edge of a clock signal, DO Sample Clock. You can configure each DIO signal to be an input, a static output, or a digital waveform generation output.

The FIFO supports a retransmit mode. In the retransmit mode, after all the samples in the FIFO have been clocked out, the FIFO begins outputting all of the samples again in the same order. For example, if the FIFO contains five samples, the pattern generated consists of sample #1, #2, #3, #4, #5, #1, #2, #3, #4, #5, #1, and so on.

The NI 6738/6739 features the following DO (waveform generation) timing signals:

- *DO Sample Clock Signal*[*]
- *DO Sample Clock Timebase Signal*
- *DO Start Trigger Signal*[*]
- *DO Pause Trigger Signal*[*]

Signals with an * support digital filtering. Refer to the *PFI Filters* section of Chapter 7, *PFI*, for more information.

DO Sample Clock Signal

The device uses the DO Sample Clock (do/SampleClock) signal to update the DO terminals with the next sample from the DO waveform generation FIFO.

You can specify an internal or external source for DO Sample Clock. You can also specify whether the digital lines update begins on the rising edge or falling edge of DO Sample Clock. If the DAQ device receives a DO Sample Clock when the FIFO is empty, the DAQ device reports an underflow error to the host software.

Using an Internal Source

One of the following internal signals can drive DO Sample Clock:

- DO Sample Clock (do/SampleClock)
- DI Sample Clock (di/SampleClock)
- Counter *n* Sample Clock
- Counter *n* Internal Output
- DI Change Detection output
- AO Sample Clock

Several other internal signals can be routed to DO Sample Clock through internal routes. Refer to *Device Routing in MAX* in the *NI-DAQmx Help* or the *LabVIEW Help* for more information.

Using an External Source

Use one of the following external signals as the source of DO Sample Clock:

- (NI PCIe/PXIe-6738) PFI <0..7>; (NI PXIe-6739) PFI <0..15>
- RTSI <0..7>
- PXI_Trig<0..7>
- PXI_STAR
- PXIe_DSTAR<A,B>

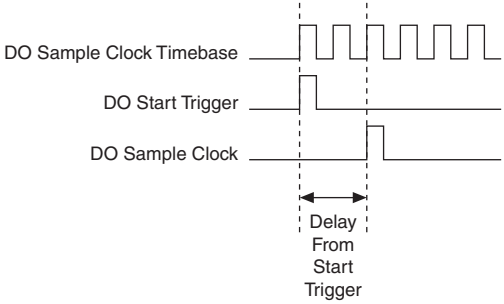
Routing DO Sample Clock to an Output Terminal

You can route DO Sample Clock (as an active low signal) out to any PFI <0..7>/<8..15>, RTSI <0..7>, PXI_Trig<0..7>, or PXIe_DSTARC terminal.

Other Timing Requirements

The DO timing engine on your device internally generates DO Sample Clock unless you select some external source. DO Start Trigger starts the timing engine and either the software or hardware can stop it once a finite generation completes. When using the DO timing engine, you can also specify a configurable delay from DO Start Trigger to the first DO Sample Clock pulse. By default, this delay is two ticks of DO Sample Clock Timebase. Figure 5-7 shows the relationship of DO Sample Clock to DO Start Trigger.

Figure 5-7. DO Sample Clock and DO Start Trigger



DO Sample Clock Timebase Signal

The DO Sample Clock Timebase (do/SampleClockTimebase) signal is divided down to provide a source for DO Sample Clock. You can route any of the following signals to be the DO Sample Clock Timebase signal:

- 100 MHz Timebase (default)
- 20 MHz Timebase
- 100 kHz Timebase
- PXI_CLK10
- (NI PCIe/PXIe-6738) PFI <0..7>; (NI PXIe-6739) PFI <0..15>
- RTSI <0..7>
- PXI_Trig<0..7>
- PXI_STAR
- PXIe_DSTAR<A,B>

DO Sample Clock Timebase is not available as an output on the I/O connector.

You might use DO Sample Clock Timebase if you want to use an external sample clock signal, but need to divide the signal down. If you want to use an external sample clock signal, but do

not need to divide the signal, then you should use DO Sample Clock rather than DO Sample Clock Timebase.

DO Start Trigger Signal

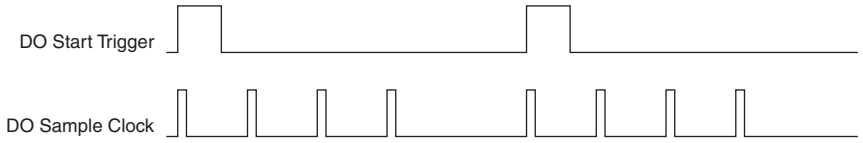
Use the DO Start Trigger (do/StartTrigger) signal to initiate a waveform generation. If you do not use triggers, you can begin a generation with a software command.

Retriggerable DO

The DO Start Trigger is configurable as retriggerable. When DO Start Trigger is configured as retriggerable, the timing engine generates the sample clocks for the configured generation in response to each pulse on a DO Start Trigger signal.

The timing engine ignores the DO Start Trigger signal while the clock generation is in progress. After the clock generation is finished, the timing engine waits for another start trigger to begin another clock generation. Figure 5-8 shows a retriggerable DO of four samples.

Figure 5-8. Retriggerable DO



Using a Digital Source

To use DO Start Trigger, specify a source and an edge. The source can be one of the following signals:

- A pulse initiated by host software
- (NI PCIe/PXIe-6738) PFI <0..7>; (NI PXIe-6739) PFI <0..15>
- RTSI <0..7>
- PXI_Trig<0..7>
- Counter *n* Internal Output
- DI Start Trigger (di/StartTrigger)
- DI Reference Trigger (di/ReferenceTrigger)
- AO Start Trigger
- Change Detection Event
- PXI_STAR
- PXIe_DSTAR<A,B>

The source can also be one of several internal signals on your DAQ device. Refer to *Device Routing in MAX* in the *NI-DAQmx Help* or the *LabVIEW Help* for more information.

You can also specify whether the waveform generation begins on the rising edge or falling edge of DO Start Trigger.

Routing DO Start Trigger Signal to an Output Terminal

You can route DO Start Trigger out to any PFI <0..7>/<8..15>, RTSI <0..7>, PXI_Trig<0..7>, or PXIe_DSTARC terminal.

The output is an active high pulse. PFI terminals are configured as inputs by default.

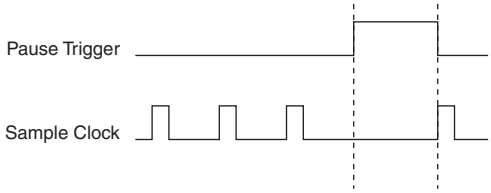
DO Pause Trigger Signal

Use the DO Pause Trigger (do/PauseTrigger) signal to mask off samples in a DAQ sequence. That is, when DO Pause Trigger is active, no samples occur.

DO Pause Trigger does not stop a sample that is in progress. The pause does not take effect until the beginning of the next sample.

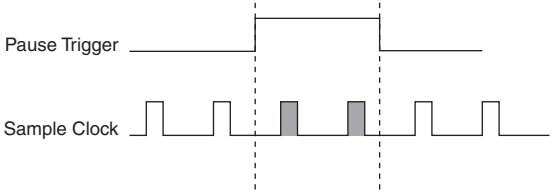
When you generate digital output signals, the generation pauses as soon as the pause trigger is asserted. If the source of your sample clock is the onboard clock, the generation resumes as soon as the pause trigger is deasserted, as shown in Figure 5-9.

Figure 5-9. DO Pause Trigger with the Onboard Clock Source



If you are using any signal other than the onboard clock as the source of your sample clock, the generation resumes as soon as the pause trigger is deasserted and another edge of the sample clock is received, as shown in Figure 5-10.

Figure 5-10. DO Pause Trigger with Other Signal Source



Using a Digital Source

To use DO Pause Trigger, specify a source and a polarity. The source can be one of the following signals:

- (NI PCIe/PXIe-6738) PFI <0..7>; (NI PXIe-6739) PFI <0..15>
- RTSI <0..7>
- PXI_Trig<0..7>
- PXI_STAR
- PXIe_DSTAR<A,B>
- Counter *n* Internal Output
- Counter *n* Gate
- DI Pause Trigger (di/PauseTrigger)
- AO Pause Trigger

The source can also be one of several other internal signals on your DAQ device. Refer to *Device Routing in MAX* in the *NI-DAQmx Help* or the *LabVIEW Help* for more information.

You can also specify whether the samples are paused when DO Pause Trigger is at a logic high or low level.

Routing DO Pause Trigger Signal to an Output Terminal

You can route DO Pause Trigger out to any PXI_Trig<0..7>, RTSI <0..7>, PFI <0..7>/<8..15>, or PXIe_DSTARC terminal.

I/O Protection

Each DIO and PFI signal is protected against overvoltage, undervoltage, and overcurrent conditions as well as ESD events. However, you should avoid these fault conditions by following these guidelines:

- If you configure a PFI or DIO line as an output, do not connect it to any external signal source, ground, or power supply.
- If you configure a PFI or DIO line as an output, understand the current requirements of the load connected to these signals. Do not exceed the specified current output limits of the DAQ device. NI has several signal conditioning solutions for digital applications requiring high current drive.
- If you configure a PFI or DIO line as an input, do not drive the line with voltages outside of its normal operating range.
- Treat the DAQ device as you would treat any static sensitive device. Always properly ground yourself and the equipment when handling the DAQ device or connecting to it.

Programmable Power-Up States

At system startup and reset, the hardware sets all PFI and DIO lines to high-impedance inputs by default. The DAQ device does not drive the signal high or low. Each line has a weak pull-down resistor connected to it, as described in the specifications document for your device.

NI-DAQmx supports programmable power-up states for PFI and DIO lines. Software can program any value at power up to the P0, P1, or P2 lines. The PFI and DIO lines can be set as:

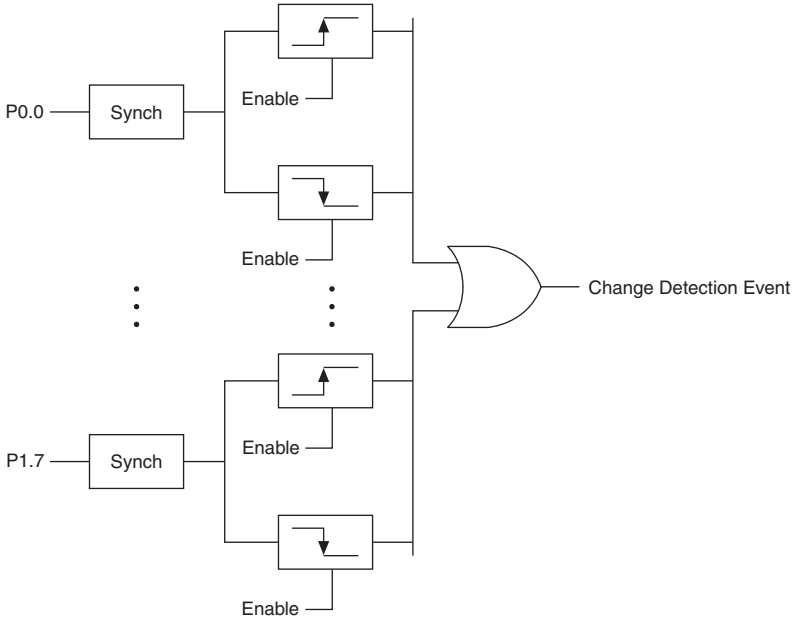
- A high-impedance input with a weak pull-down resistor (default)
- An output driving a 0
- An output driving a 1

Refer to the *NI-DAQmx Help* or the *LabVIEW Help* for more information about setting power-up states in NI-DAQmx or MAX.

DI Change Detection

You can configure the DAQ device to detect changes on all digital input lines and all PFI lines (PFI <0..7>/<8..16>/P1.<0..7>/P2.<0..7>). Figure 5-11 shows a block diagram of the DIO change detection circuitry.

Figure 5-11. DI Change Detection



You can enable the DIO change detection circuitry to detect rising edges, falling edges, or either edge individually on each DIO line. The DAQ devices synchronize each DI signal to the 100 MHz Timebase, and then sends the signal to the change detectors. The circuitry ORs the output of all enabled change detectors from every DI signal. The result of this OR is the Change Detection Event signal.

Change detection performs bus correlation by considering all changes within a 50 ns window one change detection event, which keeps signals on the same bus synchronized in samples and prevents overruns.

The Change Detection Event signal can do the following:

- Drive any RTSI <0..7>, PXI_Trig<0..7>, PFI<0..7>/PFI<8..15>, or PXI_STAR signal
- Drive the DO Sample Clock, DI Sample Clock, or AO Sample Clock
- Generate an interrupt

The Change Detection Event signal can also be used to detect changes on digital output events.

DI Change Detection Applications

The DIO change detection circuitry can interrupt a user program when one of several DIO signals changes state.

You can also use the output of the DIO change detection circuitry to trigger a DI or counter acquisition on the logical OR of several digital signals. By routing the Change Detection Event signal to a counter, you can also capture the relative time between bus changes.

You can also use the Change Detection Event signal to trigger DO, AO, or counter generations.

Digital Filtering

You can enable a programmable debouncing filter on each digital line on Port 0. When the filters are enabled, your device samples the input on each rising edge of a filter clock. The NI 6738/6739 divides down the onboard 100 MHz or 100 kHz clocks to generate the filter clock. The following is an example of low-to-high transitions of the input signal. High-to-low transitions work similarly. Refer to the *PFI Filters* section of Chapter 7, *PFI*, for more information on digital filtering on the PFI lines.

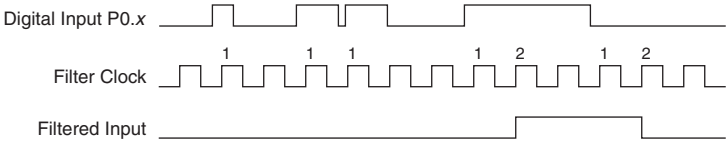
Assume that an input terminal has been low for a long time. The input terminal then changes from low-to-high, but glitches several times. When the filter clock has sampled the signal high on two consecutive edges and the signal remained stable in between, the low-to-high transition is propagated to the rest of the circuit.

Table 5-1. Filters

Filter Setting	Filter Clock	Pulse Width Guaranteed to Pass Filter	Pulse Width Guaranteed to Not Pass Filter
Short	12.5 MHz	160 ns	80 ns
Medium	195.3125 kHz	10.24 µs	5.12 µs
High	390.625 Hz	5.12 ms	2.56 ms
None	—	—	—

The filter setting for each input can be configured independently. On power up, the filters are disabled. Figure 5-12 shows an example of a low-to-high transition on an input.

Figure 5-12. Input Low-to-High Transition

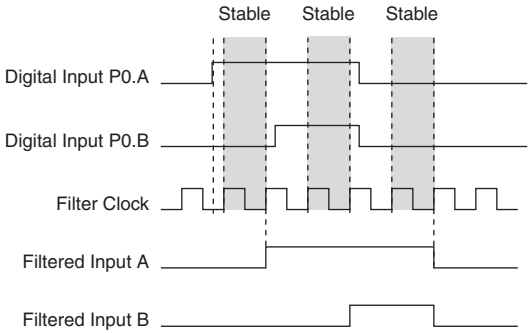


When multiple lines are configured with the same filter settings they are considered a bus. There are two filtering modes for use with multiple lines: line filtering and bus filtering. With line filtering, each line transitions independently of the other lines in the bus and acts like the behavior described above. With bus filtering, if any one line in the bus has jitter then all lines in the bus hold the state until the bus becomes stable. However, each individual line only waits one extra filter tick before changing, which prevents a noisy line from holding a valid transition indefinitely. With bus mode if all the bus line transitions become stable in less than one filter clock period and the bus period is more than two filter clock periods, then all the bus lines are guaranteed to be correlated at the output of the filter.

The behavior for each transition can be thought of as a state machine. If a line transitions and stays high for two consecutive filter clock edges, then one of two options occurs:

- **Case 1**—If no transitions have occurred on the other lines, the transition propagates on the second filtered clock edge, as shown in Figure 5-13.

Figure 5-13. Case 1



- **Case 2**—If an additional line on the bus also has a transition during the filter clock period, the change is not propagated until the next filter clock edge, as shown in Figure 5-14.

Figure 5-14. Case 2

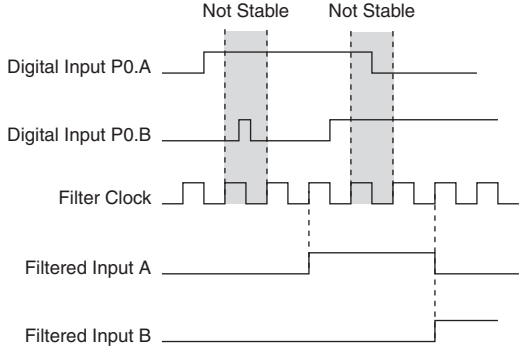
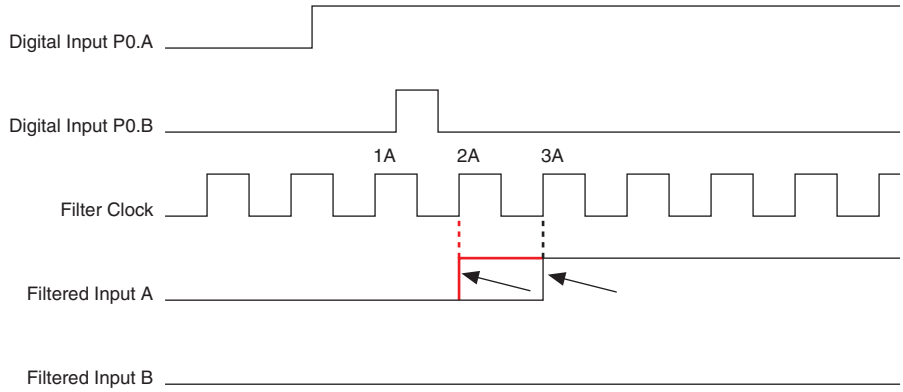


Figure 5-15 illustrates the difference between line and bus filtering.

Figure 5-15. Line and Bus Filtering



2A With line filtering, filtered input A would ignore the glitch on digital input P0.B and transition after two filter clocks.
3A Filtered input A goes high when sampled high for two consecutive filter clocks and transitions on the next filter edge because digital input P0.B glitches.

Watchdog Timer

The watchdog timer is a software-configurable feature used to set critical digital outputs to safe states in the event of a software failure, a system crash, or any other loss of communication between the application and the NI 6738/6739.

When the watchdog timer is enabled, if the NI 6738/6739 does not receive a watchdog reset software command within the time specified for the watchdog timer, the digital outputs go to a user-defined safe state and remain in that state until the watchdog timer is disarmed by the application and new values are written, the device is reset, or the computer is restarted. The expiration signal that indicates an expired watchdog will continue to assert until the watchdog is disarmed. After the watchdog timer expires, the device ignores any digital writes until the watchdog timer is disarmed.

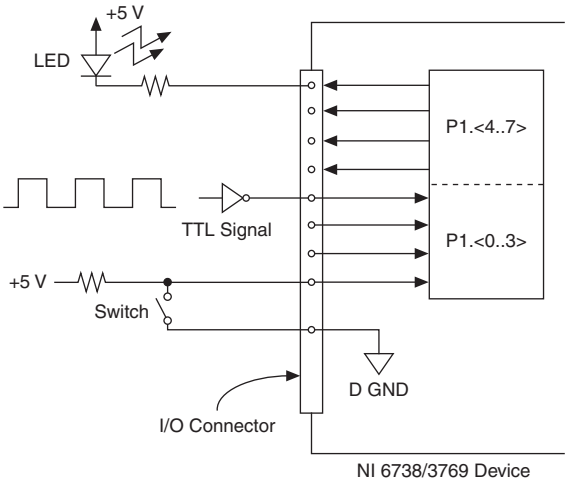
Note When the watchdog timer is enabled and the computer enters a fault condition, ports that are set to tri-state remain tri-stated and do not go to user-defined safe states.

You can set the watchdog timer timeout period to specify the amount of time that must elapse before the watchdog timer expires. The counter on the watchdog timer is configurable up to $(2^{32} - 1) \times 8$ ns (approximately 34 seconds) before it expires. A watchdog timer can be set for all DIO and PFI lines.

Connecting Digital I/O Signals

The DIO signals—P0.<0..1> and P1.<0..7> on the NI PCIe/PXIe-6738 and P0.<0..3>, P1.<0..7>, and P2.<0..7> on the NI PXIe-6739—are referenced to D GND. You can individually program each line as an input or output. Figure 5-16 shows P1.<0..3> configured for digital input and P1.<4..7> configured for digital output. Figure 5-16 shows the switch receiving TTL signals and sensing external device states and shows the LED sending TTL signals and driving external devices.

Figure 5-16. Digital I/O Connections



Caution Exceeding the maximum input voltage ratings, which are listed in the specifications document for each device, can damage the DAQ device and the computer. NI is *not* liable for any damage resulting from such signal connections.

Getting Started with DIO Applications in Software

You can use the NI 6738/6739 in the following digital I/O applications:

- Static digital input
- Static digital output
- Digital waveform generation
- Digital waveform acquisition
- DI change detection

Note For more information about programming digital I/O applications and triggers in software, refer to the *NI-DAQmx Help* or the *LabVIEW Help*.

The NI 6738/6739 uses the NI-DAQmx driver. NI-DAQmx includes a collection of programming examples to help you get started developing an application. You can modify example code and save it in an application. You can use examples to develop a new application or add example code to an existing application.

To locate LabVIEW, LabWindows/CVI, Measurement Studio, Visual Basic, and ANSI C examples, refer to the KnowledgeBase document, *Where Can I Find NI-DAQmx Examples?*, by going to `ni.com/info` and entering the Info Code `daqmxexp`.

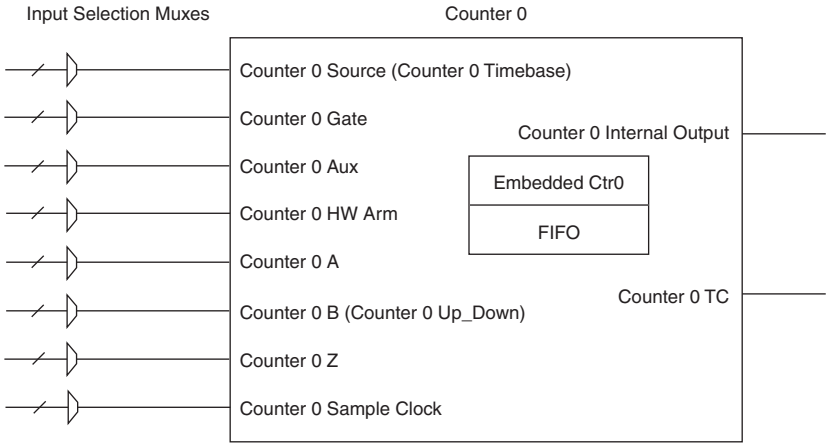
For additional examples, refer to `ni.com/examples`.

6

Counters

The NI 6738/6739 has four general-purpose 32-bit counter/timers. The general-purpose counter/timers can be used for many measurement and pulse generation applications. Figure 6-1 shows Counter 0. All four counters are identical.

Figure 6-1. NI 6738/6739 Counter 0



Counters have eight input signals, although in most applications only a few inputs are used.

For information about connecting counter signals, refer to the *Default Counter/Timer Pinouts* section.

Each counter has a FIFO that can be used for buffered acquisition and generation. Each counter also contains an embedded counter (Embedded Ctr*n*) for use in what are traditionally two-counter measurements and generations. The embedded counters cannot be programmed independent of the main counter; signals from the embedded counters are not routable.

Counter Timing Engine

Unlike analog output, digital input, and digital output, NI 6738/6739 counters do not have the ability to divide down a timebase to produce an internal counter sample clock. For sample clocked operations, an external signal must be provided to supply a clock source. The source can be any of the following signals:

- AO Sample Clock
- DI Sample Clock

- DI Start Trigger
- DO Sample Clock
- CTR *n* Internal Output
- (NI PCIe/PXIe-6738) PFI <0..7>; (NI PXIe-6739) PFI <0..15>
- PXI_Trig <0..7>
- RTSI <0..7>
- PXIe_DSTAR<A,B>
- Change Detection Event

Not all timed counter operations require a sample clock. For example, a simple buffered pulse width measurement latches in data on each edge of a pulse. For this measurement, the measured signal determines when data is latched in. These operations are referred to as implicit timed operations. However, many of the same measurements can be clocked at an interval with a sample clock. These are referred to as sample clocked operations. Table 6-1 shows the different options for the different measurements.

Note All hardware-timed single point (HWTSP) operations are sample clocked.

Table 6-1. Counter Timing Measurements

Measurement	Implicit Timing Support	Sample Clocked Timing Support
Buffered Edge Count	No	Yes
Buffered Pulse Width	Yes	Yes
Buffered Pulse	Yes	Yes
Buffered Semi-Period	Yes	No
Buffered Frequency	Yes	Yes
Buffered Period	Yes	Yes
Buffered Position	No	Yes
Buffered Two-Signal Edge Separation	Yes	Yes

Counter Input Applications

The following sections list the various counter input applications available on the NI 6738/6739:

- *Counting Edges*
- *Pulse-Width Measurement*
- *Pulse Measurement*

- *Semi-Period Measurement*
- *Frequency Measurement*
- *Period Measurement*
- *Position Measurement*
- *Two-Signal Edge-Separation Measurement*

Counting Edges

In edge counting applications, the counter counts edges on its Source after the counter is armed. You can configure the counter to count rising or falling edges on its Source input. You can also control the direction of counting (up or down), as described in the *Controlling the Direction of Counting* section. The counter values can be read on demand or with a sample clock.

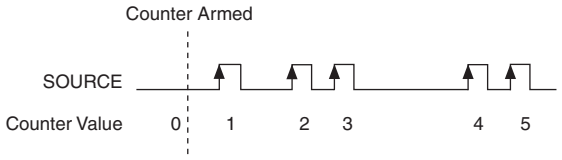
Refer to the following sections for more information about edge counting options on the NI 6738/6739:

- *Single Point (On-Demand) Edge Counting*
- *Buffered (Sample Clock) Edge Counting*

Single Point (On-Demand) Edge Counting

With single point (on-demand) edge counting, the counter counts the number of edges on the Source input after the counter is armed. On-demand refers to the fact that software can read the counter contents at any time without disturbing the counting process. Figure 6-2 shows an example of single point edge counting.

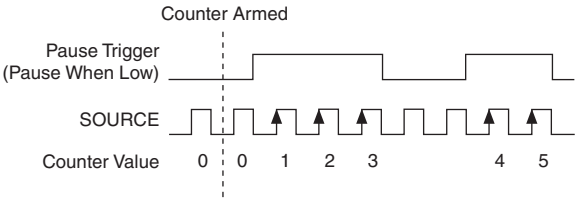
Figure 6-2. Single Point (On-Demand) Edge Counting



You can also use a pause trigger to pause (or gate) the counter. When the pause trigger is active, the counter ignores edges on its Source input. When the pause trigger is inactive, the counter counts edges normally.

You can route the pause trigger to the Gate input of the counter. You can configure the counter to pause counting when the pause trigger is high or when it is low. Figure 6-3 shows an example of on-demand edge counting with a pause trigger.

Figure 6-3. Single Point (On-Demand) Edge Counting with Pause Trigger



Buffered (Sample Clock) Edge Counting

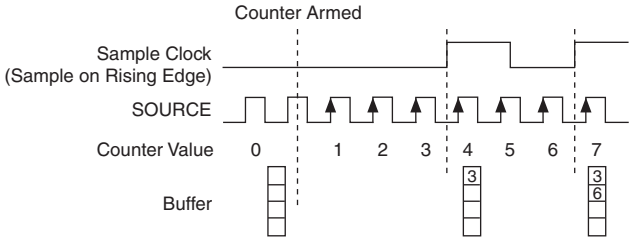
With buffered edge counting (edge counting using a sample clock), the counter counts the number of edges on the Source input after the counter is armed. The value of the counter is sampled on each active edge of a sample clock and stored in the FIFO. A DMA controller transfers the sampled values to host memory.

The count values returned are the cumulative counts since the counter armed event. That is, the sample clock does not reset the counter.

You can configure the counter to sample on the rising or falling edge of the sample clock.

Figure 6-4 shows an example of buffered edge counting. Notice that counting begins when the counter is armed, which occurs before the first active edge on Sample Clock.

Figure 6-4. Buffered (Sample Clock) Edge Counting



Controlling the Direction of Counting

In edge counting applications, the counter can count up or down. You can configure the counter to do the following:

- Always count up
- Always count down
- Count up when the Counter 0 B input is high; count down when it is low

For information about connecting counter signals, refer to the *Default Counter/Timer Pinouts* section.

Pulse-Width Measurement

In pulse-width measurements, the counter measures the width of a pulse on its Gate input signal. You can configure the counter to measure the width of high pulses or low pulses on the Gate signal.

You can route an internal or external periodic clock signal (with a known period) to the Source input of the counter. The counter counts the number of rising (or falling) edges on the Source signal while the pulse on the Gate signal is active.

You can calculate the pulse width by multiplying the period of the Source signal by the number of edges returned by the counter.

A pulse-width measurement is accurate even if the counter is armed while a pulse train is in progress. If a counter is armed while the pulse is in the active state, it waits for the next transition to the active state to begin the measurement.

Refer to the following sections for more information about pulse-width measurement options on the NI 6738/6739:

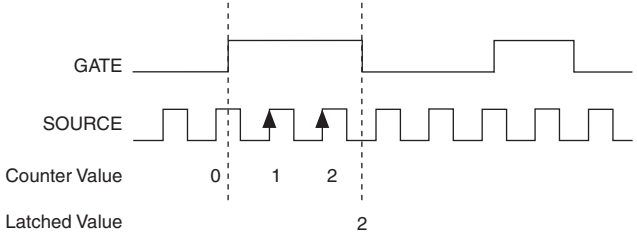
- *Single Pulse-Width Measurement*
- *Implicit Buffered Pulse-Width Measurement*
- *Sample Clocked Buffered Pulse-Width Measurement*
- *Hardware-Timed Single Point Pulse-Width Measurement*

Single Pulse-Width Measurement

With single pulse-width measurement, the counter counts the number of edges on the Source input while the Gate input remains active. When the Gate input goes inactive, the counter stores the count in the FIFO and ignores other edges on the Gate and Source inputs. Software then reads the stored count.

Figure 6-5 shows an example of a single pulse-width measurement.

Figure 6-5. Single Pulse-Width Measurement



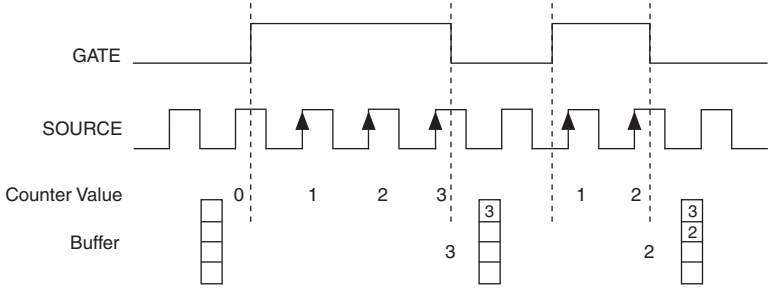
Implicit Buffered Pulse-Width Measurement

An implicit buffered pulse-width measurement is similar to single pulse-width measurement, but buffered pulse-width measurement takes measurements over multiple pulses.

The counter counts the number of edges on the Source input while the Gate input remains active. On each trailing edge of the Gate signal, the counter stores the count in the counter FIFO. A DMA controller transfers the stored values to host memory.

Figure 6-6 shows an example of an implicit buffered pulse-width measurement.

Figure 6-6. Implicit Buffered Pulse-Width Measurement



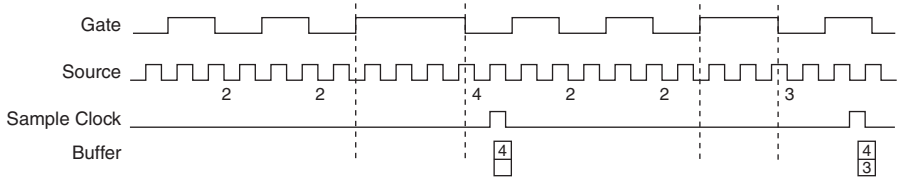
Sample Clocked Buffered Pulse-Width Measurement

A Sample Clocked Buffered pulse-width measurement is similar to single pulse-width measurement, but buffered pulse-width measurement takes measurements over multiple pulses correlated to a sample clock.

The counter counts the number of edges on the Source input while the Gate input remains active. On each sample clock edge, the counter stores the count in the FIFO of the last pulse width to complete. A DMA controller transfers the stored values to host memory.

Figure 6-7 shows an example of a sample clocked buffered pulse-width measurement.

Figure 6-7. Sample Clocked Buffered Pulse-Width Measurement



Hardware-Timed Single Point Pulse-Width Measurement

A hardware-timed single point (HWTSP) pulse-width measurement has the same behavior as a sample clocked buffered pulse-width measurement.

Note If a pulse does not occur between sample clocks, an overrun error occurs.

For information about connecting counter signals, refer to the *Default Counter/Timer Pinouts* section.

Pulse Measurement

In pulse measurements, the counter measures the high and low time of a pulse on its Gate input signal after the counter is armed. A pulse is defined in terms of its high and low time, high and low ticks or frequency and duty cycle, which is similar to the pulse-width measurement, except that the inactive pulse is measured as well.

You can route an internal or external periodic clock signal (with a known period) to the Source input of the counter. The counter counts the number of rising (or falling) edges occurring on the Source input between two edges of the Gate signal.

You can calculate the high and low time of the Gate input by multiplying the period of the Source signal by the number of edges returned by the counter.

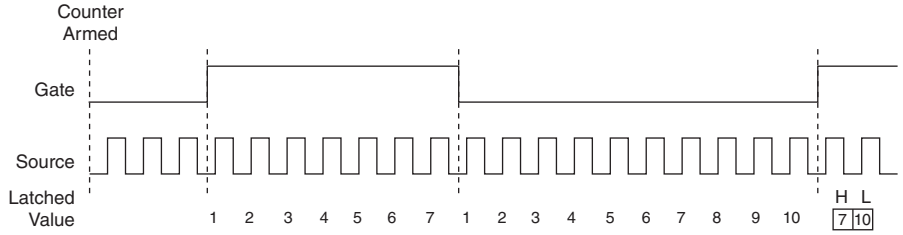
Refer to the following sections for more information about pulse measurement options on the NI 6738/6739:

- *Single Pulse Measurement*
- *Implicit Buffered Pulse Measurement*
- *Sample Clocked Buffered Pulse Measurement*
- *Hardware-Timed Single Point Pulse Measurement*

Single Pulse Measurement

Single (on-demand) pulse measurement is equivalent to two single pulse-width measurements on the high (H) and low (L) ticks of a pulse, as shown in Figure 6-8.

Figure 6-8. Single (On-Demand) Pulse Measurement



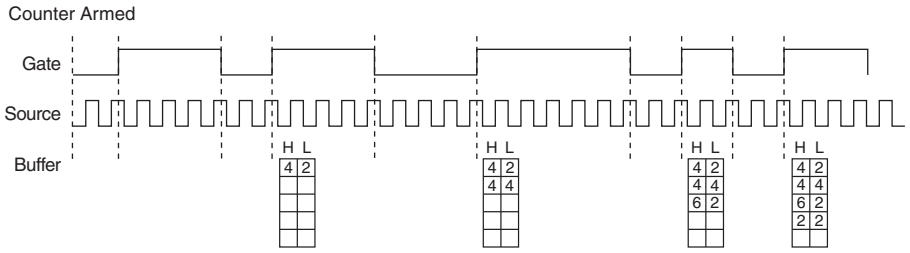
Implicit Buffered Pulse Measurement

In an implicit buffered pulse measurement, on each edge of the Gate signal, the counter stores the count in the FIFO. A DMA controller transfers the stored values to host memory.

The counter begins counting when it is armed. The arm usually occurs between edges on the Gate input, but the counting does not start until the desired edge. You can select whether to read the high pulse or low pulse first using the StartingEdge property in NI-DAQmx.

Figure 6-9 shows an example of an implicit buffered pulse measurement.

Figure 6-9. Implicit Buffered Pulse Measurement



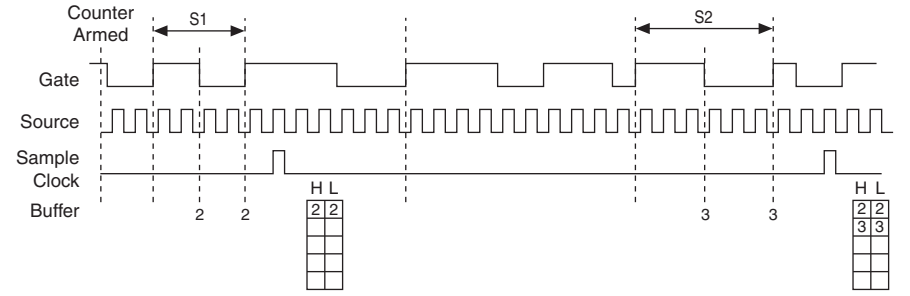
Sample Clocked Buffered Pulse Measurement

A sample clocked buffered pulse measurement is similar to single pulse measurement, but a buffered pulse measurement takes measurements over multiple pulses correlated to a sample clock.

The counter performs a pulse measurement on the Gate. On each sample clock edge, the counter stores the high and low ticks in the FIFO of the last pulse to complete. A DMA controller transfers the stored values to host memory.

Figure 6-10 shows an example of a sample clocked buffered pulse measurement.

Figure 6-10. Sample Clocked Buffered Pulse Measurement



Hardware-Timed Single Point Pulse Measurement

A hardware-timed single point (HWTSP) pulse measurement has the same behavior as a sample clocked buffered pulse measurement.

Note If a pulse does not occur between sample clocks, an overrun error occurs.

For information about connecting counter signals, refer to the *Default Counter/Timer Pinouts* section.

Pulse versus Semi-Period Measurements

In hardware, pulse measurement and semi-period are the same measurement. Both measure the high and low times of a pulse. The functional difference between the two measurements is how the data is returned. In a semi-period measurement, each high or low time is considered one point of data and returned in units of seconds or ticks. In a pulse measurement, each pair of high and low times is considered one point of data and returned as a paired sample in units of frequency and duty cycle, high and low time or high and low ticks. When reading data, 10 points in a semi-period measurement gets an array of five high times and five low times. When you read 10 points in a pulse measurement, you get an array of 10 pairs of high and low times.

Also, pulse measurements support sample clock timing while semi-period measurements do not.

Semi-Period Measurement

In semi-period measurements, the counter measures a semi-period on its Gate input signal after the counter is armed. A semi-period is the time between any two consecutive edges on the Gate input.

You can route an internal or external periodic clock signal (with a known period) to the Source input of the counter. The counter counts the number of rising (or falling) edges occurring on the Source input between two edges of the Gate signal.

You can calculate the semi-period of the Gate input by multiplying the period of the Source signal by the number of edges returned by the counter.

Refer to the following sections for more information about semi-period measurement options on the NI 6738/6739:

- *Single Semi-Period Measurement*
- *Implicit Buffered Semi-Period Measurement*

Refer to the *Pulse versus Semi-Period Measurements* section for information about the differences between semi-period measurement and pulse measurement.

Single Semi-Period Measurement

Single semi-period measurement is equivalent to single pulse-width measurement.

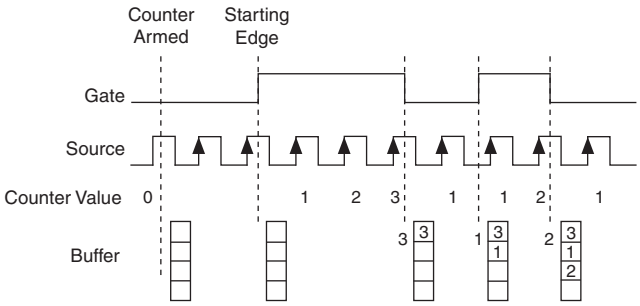
Implicit Buffered Semi-Period Measurement

In implicit buffered semi-period measurement, on each edge of the Gate signal, the counter stores the count in the FIFO. A DMA controller transfers the stored values to host memory.

The counter begins counting when it is armed. The arm usually occurs between edges on the Gate input. You can select whether to read the first active low or active high semi period using the CI.SemiPeriod.StartingEdge property in NI-DAQmx.

Figure 6-11 shows an example of an implicit buffered semi-period measurement.

Figure 6-11. Implicit Buffered Semi-Period Measurement



For information about connecting counter signals, refer to the *Default Counter/Timer Pinouts* section.

Frequency Measurement

You can use the counters to measure frequency in several different ways. Refer to the following sections for information about frequency measurement options on the NI 6738/6739:

- *Low Frequency with One Counter*
- *High Frequency with Two Counters*
- *Large Range of Frequencies with Two Counters*
- *Sample Clocked Buffered Frequency Measurement*
- *Hardware-Timed Single Point Frequency Measurement*

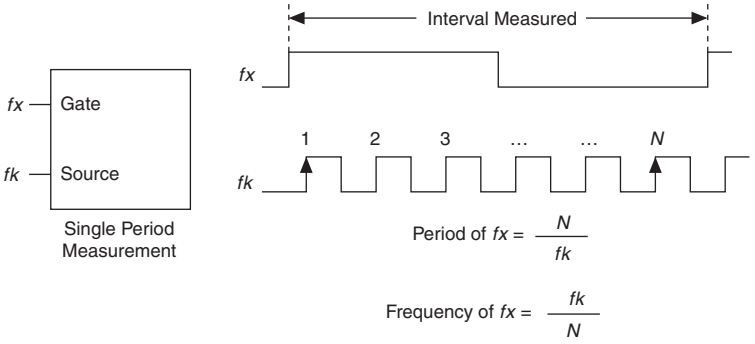
Low Frequency with One Counter

For low frequency measurements with one counter, you measure one period of your signal using a known timebase.

You can route the signal to measure (*fx*) to the Gate of a counter. You can route a known timebase (*fk*) to the Source of the counter. The known timebase can be an onboard timebase, such as 100 MHz Timebase, 20 MHz Timebase, or 100 kHz Timebase, or any other signal with a known rate.

You can configure the counter to measure one period of the gate signal. The frequency of *fx* is the inverse of the period. Figure 6-12 illustrates this method.

Figure 6-12. Low Frequency with One Counter



High Frequency with Two Counters

For high frequency measurements with two counters, you measure one pulse of a known width using your signal and derive the frequency of your signal from the result.

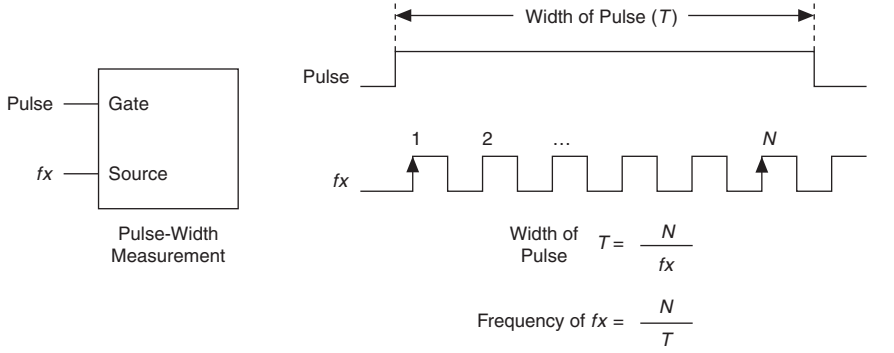
Note Counter 0 is always paired with Counter 1. Counter 2 is always paired with Counter 3.

In this method, you route a pulse of known duration (*T*) to the Gate of a counter. You can generate the pulse using a second counter. You can also generate the pulse externally and connect it to a PFI or PXI_Trig terminal. You only need to use one counter if you generate the pulse externally.

Route the signal to measure (*fx*) to the Source of the counter. Configure the counter for a single pulse-width measurement. If you measure the width of pulse *T* to be *N* periods of *fx*, the frequency of *fx* is *N*/*T*.

Figure 6-13 illustrates this method. Another option is to measure the width of a known period instead of a known pulse.

Figure 6-13. High Frequency with Two Counters



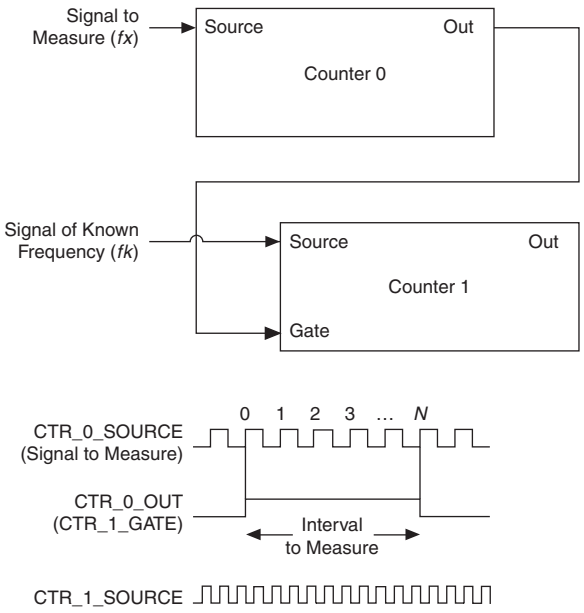
Large Range of Frequencies with Two Counters

By using two counters, you can accurately measure a signal that might be high or low frequency. This technique is called reciprocal frequency measurement. When measuring a large range of frequencies with two counters, you generate a long pulse using the signal to measure. You then measure the long pulse with a known timebase. The NI 6738/6739 can measure this long pulse more accurately than the faster input signal.

Note Counter 0 is always paired with Counter 1. Counter 2 is always paired with Counter 3.

You can route the signal to measure to the Source input of Counter 0, as shown in Figure 6-14. Assume this signal to measure has frequency *fx*. NI-DAQmx automatically configures Counter 0 to generate a single pulse that is the width of *N* periods of the source input signal.

Figure 6-14. Large Range of Frequencies with Two Counters



NI-DAQmx then routes the Counter 0 Internal Output signal to the gate of Counter 1. You can then route a signal of known frequency (*fk*) as a counter timebase to the Counter 1 Source input. NI-DAQmx configures Counter 1 to perform a single pulse-width measurement. Suppose the result is that the pulse width is *J* periods of the *fk* clock.

From Counter 0, the length of the pulse is *N*/*fx*. From Counter 1, the length of the same pulse is *J*/*fk*. Therefore, the frequency of *fx* is given by *fx* = *fk* * (*N*/*J*).

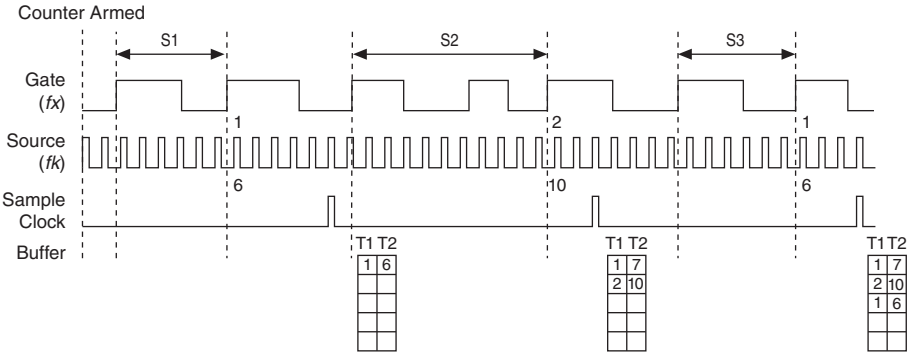
Sample Clocked Buffered Frequency Measurement

Sample clocked buffered point frequency measurements can either be a single frequency measurement or an average between sample clocks. Use CI.Freq.EnableAveraging to set the behavior. For buffered frequency, the default is True. For hardware-timed single point (HWTSP), the default is False.

A sample clocked buffered frequency measurement with CI.Freq.EnableAveraging set to True uses the embedded counter and a sample clock to perform a frequency measurement. For each sample clock period, the embedded counter counts the signal to measure (*fx*) and the primary counter counts the internal time-base of a known frequency (*fk*). Suppose T1 is the number of

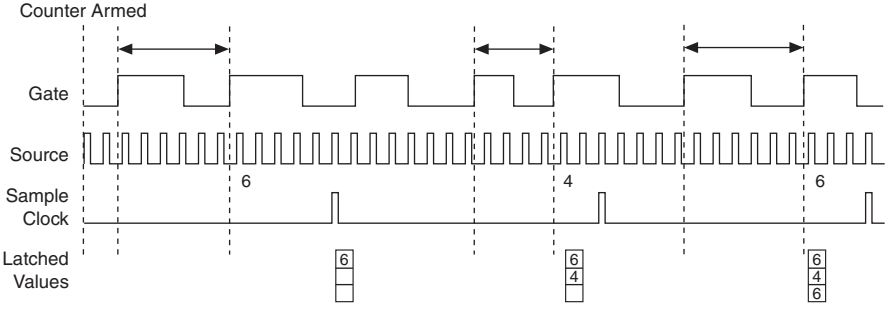
ticks of the unknown signal counted between sample clocks and T2 is the number of ticks counted of the known time-base. The frequency measured will be $fx = fk * (T1/T2)$.

Figure 6-15. Sample Clocked Buffered Frequency Measurement (Averaging)



When CI.Freq.EnableAveraging is set to false, the frequency measurement returns the frequency of the pulse just before the sample clock. This single measurement is a single frequency measurement and is not an average between clocks.

Figure 6-16. Sample Clocked Buffered Frequency Measurement (Non-Averaging)

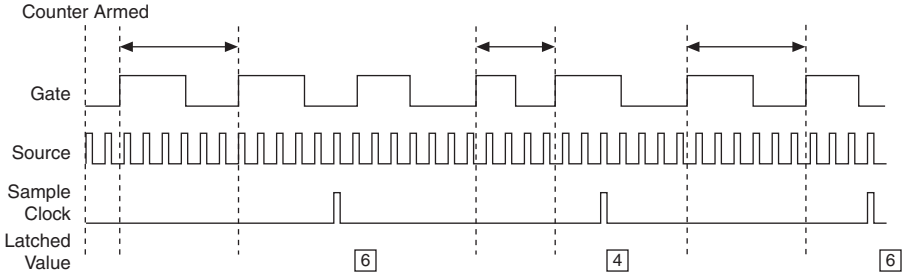


With sample clocked frequency measurements, ensure that the frequency to measure is twice as fast as the sample clock to prevent a measurement overflow.

Hardware-Timed Single Point Frequency Measurement

Hardware-timed single point (HWTSP) frequency measurements can either be a single frequency measurement or an average between sample clocks. Use CI.Freq.EnableAveraging to set the behavior. For hardware-timed single point, the default is False. Refer to the *Sample Clocked Buffered Frequency Measurement* section for more information.

Figure 6-17. Hardware-Timed Single Point Frequency Measurement



Choosing a Method for Measuring Frequency

The best method to measure frequency depends on several factors including the expected frequency of the signal to measure, the desired accuracy, how many counters are available, and how long the measurement can take. For all frequency measurement methods, assume the following:

fx	is the frequency to be measured if no error
fk	is the known source or gate frequency
measurement time (*T*)	is the time it takes to measure a single sample
Divide down (*N*)	is the integer to divide down measured frequency, only used in large range two counters
fs	is the sample clock rate, only used in sample clocked frequency measurements

Here is how these variables apply to each method, summarized in Table 6-2.

- **One counter**—With one counter measurements, a known timebase is used for the source frequency (*fk*). The measurement time is the period of the frequency to be measured, or 1/*fx*.
- **Two counter high frequency**—With the two counter high frequency method, the second counter provides a known measurement time. The gate frequency equals 1/*measurement time*.
- **Two counter large range**—The two counter larger range measurement is the same as a one counter measurement, but now the user has an integer divide down of the signal. An internal

timebase is still used for the source frequency (*fk*), but the divide down means that the measurement time is the period of the divided down signal, or *N*/*fx* where *N* is the divide down.

- **Sample clocked**—For sample clocked frequency measurements, a known timebase is counted for the source frequency (*fk*). The measurement time is the period of the sample clock (*fs*).

Table 6-2. Frequency Measurement Methods

Variable	Sample Clocked	One Counter	Two Counter	
			High Frequency	Large Range
fk	Known timebase	Known timebase	$\frac{1}{\text{gating period}}$	Known timebase
Measurement time	$\frac{1}{fs}$	$\frac{1}{fx}$	gating period	$\frac{N}{fx}$
Max. frequency error	$fx \times \frac{fx}{fk \times \left\lfloor \frac{fx}{fs} - 1 \right\rfloor}$	$fx \times \frac{fx}{fk - fx}$	*fk*	$fx \times \frac{fx}{N \times fk - fx}$
Max. error %	$\frac{fx}{fk \times \left\lfloor \frac{fx}{fs} - 1 \right\rfloor}$	$\frac{fx}{fk - fx}$	$\frac{fk}{fx}$	$\frac{fx}{N \times fk - fx}$
Note: Accuracy equations do not take clock stability into account. Refer to your device specifications for clock stability.				

Which Method Is Best?

This depends on the frequency to be measured, the rate at which you want to monitor the frequency and the accuracy you desire. Take for example, measuring a 50 kHz signal. Assuming that the measurement times for the sample clocked (with averaging) and two counter frequency measurements are configured the same, Table 6-3 summarizes the results.

Table 6-3. 50 kHz Frequency Measurement Methods

Variable	Sample Clocked	One Counter	Two Counter	
			High Frequency	Large Range
fx	50,000	50,000	50,000	50,000
fk	100 M	100 M	1,000	100 M

Table 6-3. 50 kHz Frequency Measurement Methods (Continued)

Variable	Sample Clocked	One Counter	Two Counter	
			High Frequency	Large Range
Measurement time (*mS*)	1	.02	1	1
N	—	—	—	50
Max. frequency error (Hz)	.512	25	1,000	.5
Max. error %	.00102	.05	2	.001

From these results, you can see that while the measurement time for one counter is shorter, the accuracy is best in the sample clocked and two counter large range measurements. For another example, Table 6-4 shows the results for 5 MHz.

Table 6-4. 5 MHz Frequency Measurement Methods

Variable	Sample Clocked	One Counter	Two Counter	
			High Frequency	Large Range
fx	5 M	5 M	5 M	5 M
fk	100 M	100 M	1,000	100 M
Measurement time (*mS*)	1	.0002	1	1
N	—	—	—	5,000
Max. Frequency error (Hz)	50.01	263 k	1,000	50
Max. Error %	.001	5.26	.02	.001

Again the measurement time for the one counter measurement is lowest, but the accuracy is lower. Note that the accuracy and measurement time of the sample clocked and two counter large range are almost the same. The advantage of the sample clocked method is that even when the frequency to measure changes, the measurement time does not and error percentage varies little. For example, if you configured a large range two counter measurement to use a divide down of 50 for a 50 k signal, then you would get the accuracy measurement time and accuracy listed in Table 6-3. But if your signal ramped up to 5 M, then with a divide down of 50, your measurement time is 0.01 ms, but your error is now 0.1%. The error with a sample clocked frequency

measurement is not as dependent on the measured frequency so at 50 k and 5 M with a measurement time of 1 ms the error percentage is still close to 0.001%. One of the disadvantages of a sample clocked frequency measurement is that the frequency to be measured must be at least twice the sample clock rate to ensure that a full period of the frequency to be measured occurs between sample clocks.

- Low frequency measurements with one counter is a good method for many applications. However, the accuracy of the measurement decreases as the frequency increases.
- High frequency measurements with two counters is accurate for high frequency signals. However, the accuracy decreases as the frequency of the signal to measure decreases. At very low frequencies, this method may be too inaccurate for your application. Another disadvantage of this method is that it requires two counters (if you cannot provide an external signal of known width). An advantage of high frequency measurements with two counters is that the measurement completes in a known amount of time.
- Measuring a large range of frequencies with two counters measures high and low frequency signals accurately. However, it requires two counters, and it has a variable sample time and variable error % dependent on the input signal.
- Again, the measurement time for the one counter measurement is lowest, but the accuracy is lower. Note that the accuracy and measurement time of the sample clocked and two counter large range are the same. The advantage of the sample clocked method is that even when the frequency to measure changes, the measurement time and error % does not. For example, if you configured a large range two counter measurement to use a divide down of 50 for a 50 kHz signal, then you would get the accuracy measurement time and accuracy listed in table 6-3. But if your signal ramped up to 5 MHz, then with a divide down of 50, your measurement time would be 0.01 ms, but your error would now be 0.001%. The error with a sample clocked frequency measurement is not dependent on the measured frequency so at 50 kHz and 5 MHz with a measurement time of 1 ms the error % will still be 0.001%. One of the disadvantages of a sample clocked frequency measurement is that the frequency to be measured must be at least twice the sample clock rate to ensure that a full period of the frequency to be measured occurs between sample clocks.

Table 6-5 summarizes some of the differences in methods of measuring frequency.

Table 6-5. Frequency Measurement Method Comparison

Method	Number of Counters Used	Number of Measurements Returned	Measures High Frequency Signals Accurately	Measures Low Frequency Signals Accurately
Low frequency with one counter	1	1	Poor	Good
High frequency with two counters	1 or 2	1	Good	Poor

Table 6-5. Frequency Measurement Method Comparison (Continued)

Method	Number of Counters Used	Number of Measurements Returned	Measures High Frequency Signals Accurately	Measures Low Frequency Signals Accurately
Large range of frequencies with two counters	2	1	Good	Good
Sample clocked (averaged)	1	1	Good	Good

For information about connecting counter signals, refer to the *Default Counter/Timer Pinouts* section.

Period Measurement

In period measurements, the counter measures a period on its Gate input signal after the counter is armed. You can configure the counter to measure the period between two rising edges or two falling edges of the Gate input signal.

You can route an internal or external periodic clock signal (with a known period) to the Source input of the counter. The counter counts the number of rising (or falling) edges occurring on the Source input between the two active edges of the Gate signal.

You can calculate the period of the Gate input by multiplying the period of the Source signal by the number of edges returned by the counter.

Period measurements return the inverse results of frequency measurements. Refer to the *Frequency Measurement* section for more information.

Position Measurement

You can use the counters to perform position measurements with quadrature encoders or two-pulse encoders. You can measure angular position with X1, X2, and X4 angular encoders. Linear position can be measured with two-pulse encoders. You can choose to do either a single point (on-demand) position measurement or a buffered (sample clock) position measurement. You must arm a counter to begin position measurements.

Refer to the following sections for more information about the position measurement options on the NI 6738/6739:

- *Measurements Using Quadrature Encoders*
- *Measurements Using Two Pulse Encoders*
- *Buffered (Sample Clock) Position Measurement*

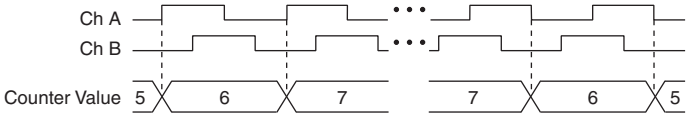
Measurements Using Quadrature Encoders

The counters can perform measurements of quadrature encoders that use X1, X2, or X4 encoding. A quadrature encoder can have up to three channels—channels A, B, and Z.

- **X1 Encoding**—When channel A leads channel B in a quadrature cycle, the counter increments. When channel B leads channel A in a quadrature cycle, the counter decrements. The amount of increments and decrements per cycle depends on the type of encoding—X1, X2, or X4.

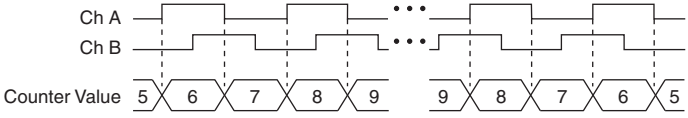
  Figure 6-18 shows a quadrature cycle and the resulting increments and decrements for X1 encoding. When channel A leads channel B, the increment occurs on the rising edge of channel A. When channel B leads channel A, the decrement occurs on the falling edge of channel A.

Figure 6-18. X1 Encoding



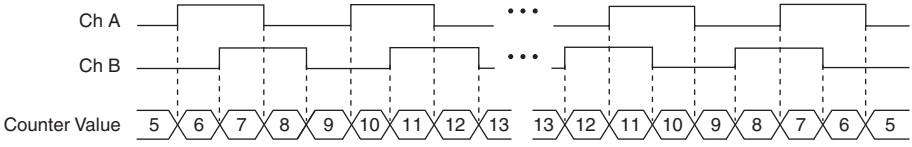
- **X2 Encoding**—The same behavior holds for X2 encoding except the counter increments or decrements on each edge of channel A, depending on which channel leads the other. Each cycle results in two increments or decrements, as shown in Figure 6-19.

Figure 6-19. X2 Encoding



- **X4 Encoding**—Similarly, the counter increments or decrements on each edge of channels A and B for X4 encoding. Whether the counter increments or decrements depends on which channel leads the other. Each cycle results in four increments or decrements, as shown in Figure 6-20.

Figure 6-20. X4 Encoding



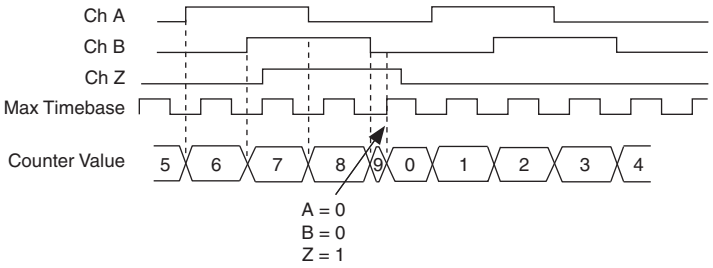
Channel Z Behavior

Some quadrature encoders have a third channel, channel Z, which is also referred to as the index channel. A high level on channel Z causes the counter to be reloaded with a specified value in a specified phase of the quadrature cycle. You can program the counter reload to occur in any one of the four phases in a quadrature cycle.

Channel Z behavior—when it goes high and how long it stays high—differs with quadrature encoder designs. You must refer to the documentation for your quadrature encoder to obtain timing of channel Z with respect to channels A and B. You must then ensure that channel Z is high during at least a portion of the phase you specify for reload. For instance, in Figure 6-21, channel Z is never high when channel A is high and channel B is low. Thus, the reload must occur in some other phase.

In Figure 6-21, the reload phase is when both channel A and channel B are low. The reload occurs when the phase is true and channel Z is high. Incrementing and decrementing takes priority over reloading. Thus, when the channel B goes low to enter the reload phase, the increment occurs first. The reload occurs within one maximum timebase period after the reload phase becomes true. After the reload occurs, the counter continues to count as before. Figure 6-21 illustrates channel Z reload with X4 decoding.

Figure 6-21. Channel Z Reload with X4 Decoding

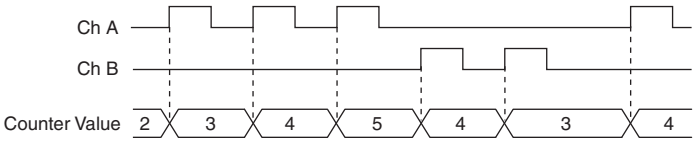


Measurements Using Two Pulse Encoders

The counter supports two pulse encoders that have two channels—channels A and B.

The counter increments on each rising edge of channel A. The counter decrements on each rising edge of channel B, as shown in Figure 6-22.

Figure 6-22. Measurements Using Two Pulse Encoders



For information about connecting counter signals, refer to the *Default Counter/Timer Pinouts* section.

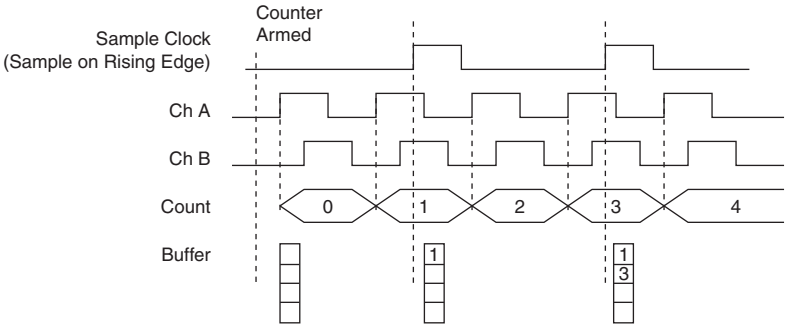
Buffered (Sample Clock) Position Measurement

With buffered position measurement (position measurement using a sample clock), the counter increments based on the encoding used after the counter is armed. The value of the counter is sampled on each active edge of a sample clock. A DMA controller transfers the sampled values

to host memory. The count values returned are the cumulative counts since the counter armed event; that is, the sample clock does not reset the counter. You can route the counter sample clock to the Gate input of the counter. You can configure the counter to sample on the rising or falling edge of the sample clock.

Figure 6-23 shows an example of a buffered X1 position measurement.

Figure 6-23. Buffered Position Measurement



Hardware-Timed Single Point Position Measurement

A hardware-timed single point (HWTSP) position measurement has the same behavior as a buffered (sample clock) position measurement.

For information about connecting counter signals, refer to the *Default Counter/Timer Pinouts* section.

Two-Signal Edge-Separation Measurement

Two-signal edge-separation measurement is similar to pulse-width measurement, except that there are two measurement signals—Aux and Gate. An active edge on the Aux input starts the counting and an active edge on the Gate input stops the counting. You must arm a counter to begin a two edge separation measurement.

After the counter has been armed and an active edge occurs on the Aux input, the counter counts the number of rising (or falling) edges on the Source. The counter ignores additional edges on the Aux input.

The counter stops counting upon receiving an active edge on the Gate input. The counter stores the count in the FIFO.

You can configure the rising or falling edge of the Aux input to be the active edge. You can configure the rising or falling edge of the Gate input to be the active edge.

Use this measurement type to count events or measure the time that occurs between edges on two signals. This type of measurement is sometimes referred to as start/stop trigger measurement, second gate measurement, or A-to-B measurement.

Refer to the following sections for more information about the edge-separation measurement options on the NI 6738/6739:

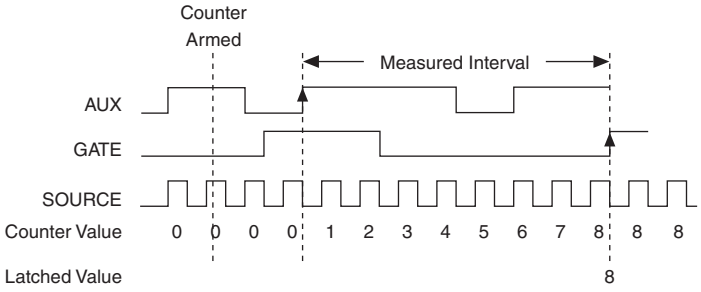
- *Single Two-Signal Edge-Separation Measurement*
- *Implicit Buffered Two-Signal Edge-Separation Measurement*
- *Sample Clocked Buffered Two-Signal Separation Measurement*
- *Hardware-Timed Single Point Two-Signal Separation Measurement*

Single Two-Signal Edge-Separation Measurement

With single two-signal edge-separation measurement, the counter counts the number of rising (or falling) edges on the Source input occurring between an active edge of the Gate signal and an active edge of the Aux signal. The counter then stores the count in the FIFO and ignores other edges on its inputs. Software then reads the stored count.

Figure 6-24 shows an example of a single two-signal edge-separation measurement.

Figure 6-24. Single Two-Signal Edge-Separation Measurement



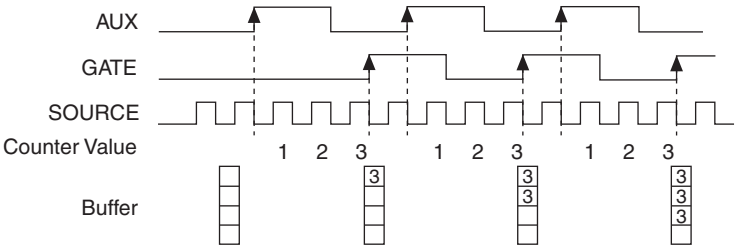
Implicit Buffered Two-Signal Edge-Separation Measurement

Implicit buffered and single two-signal edge-separation measurements are similar, but implicit buffered measurement measures multiple intervals.

The counter counts the number of rising (or falling) edges on the Source input occurring between an active edge of the Gate signal and an active edge of the Aux signal. The counter then stores the count in the FIFO. On the next active edge of the Gate signal, the counter begins another measurement. A DMA controller transfers the stored values to host memory.

Figure 6-25 shows an example of an implicit buffered two-signal edge-separation measurement.

Figure 6-25. Implicit Buffered Two-Signal Edge-Separation Measurement

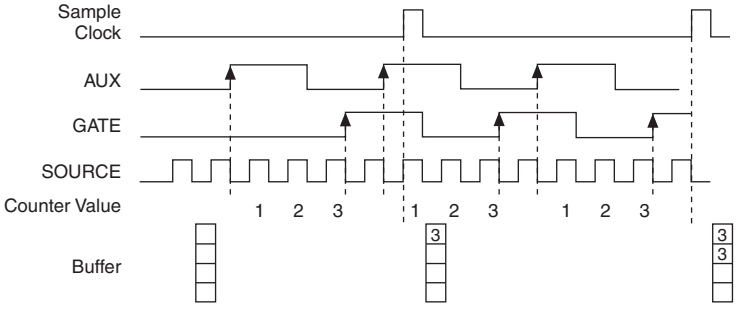


Sample Clocked Buffered Two-Signal Separation Measurement

A sample clocked buffered two-signal separation measurement is similar to single two-signal separation measurement, but buffered two-signal separation measurement takes measurements over multiple intervals correlated to a sample clock. The counter counts the number of rising (or falling) edges on the Source input occurring between an active edge of the Gate signal and an active edge of the Aux signal. The counter then stores the count in the FIFO on a sample clock edge. On the next active edge of the Gate signal, the counter begins another measurement. A DMA controller transfers the stored values to host memory.

Figure 6-26 shows an example of a sample clocked buffered two-signal separation measurement.

Figure 6-26. Sample Clocked Buffered Two-Signal Separation Measurement



Hardware-Timed Single Point Two-Signal Separation Measurement

A hardware-timed single point (HWTSP) two-signal separation measurement has the same behavior as a sample clocked buffered two-signal separation measurement. Refer to the *Sample Clocked Buffered Two-Signal Separation Measurement* section for more information.

Note If an active edge on the Gate and an active edge on the AUX does not occur between sample clocks, an overrun error occurs.

For information about connecting counter signals, refer to the *Default Counter/Timer Pinouts* section.

Counter Output Applications

The following sections list the various counter output applications available on the NI 6738/6739:

- *Simple Pulse Generation*
- *Pulse Train Generation*
- *Frequency Generation*
- *Frequency Division*
- *Pulse Generation for ETS*

Simple Pulse Generation

Refer to the following sections for more information about the simple pulse generation options on the NI 6738/6739:

- *Single Pulse Generation*
- *Single Pulse Generation with Start Trigger*

Single Pulse Generation

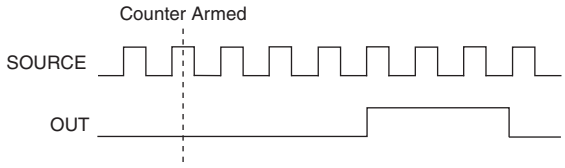
The counter can output a single pulse. The pulse appears on the Counter *n* Internal Output signal of the counter.

You can specify a delay from when the counter is armed to the beginning of the pulse. The delay is measured in terms of a number of active edges of the Source input.

You can specify a pulse width. The pulse width is also measured in terms of a number of active edges of the Source input. You can also specify the active edge of the Source input (rising or falling).

Figure 6-27 shows a generation of a pulse with a pulse delay of four and a pulse width of three (using the rising edge of Source).

Figure 6-27. Single Pulse Generation



Single Pulse Generation with Start Trigger

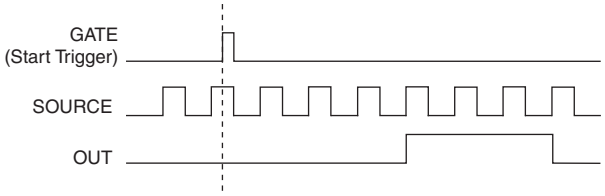
The counter can output a single pulse in response to one pulse on a hardware Start Trigger signal. The pulse appears on the Counter *n* Internal Output signal of the counter.

You can route the Start Trigger signal to the Gate input of the counter. You can specify a delay from the Start Trigger to the beginning of the pulse. You can also specify the pulse width. The delay and pulse width are measured in terms of a number of active edges of the Source input.

After the Start Trigger signal pulses once, the counter ignores the Gate input.

Figure 6-28 shows a generation of a pulse with a pulse delay of four and a pulse width of three (using the rising edge of Source).

Figure 6-28. Single Pulse Generation with Start Trigger



Pulse Train Generation

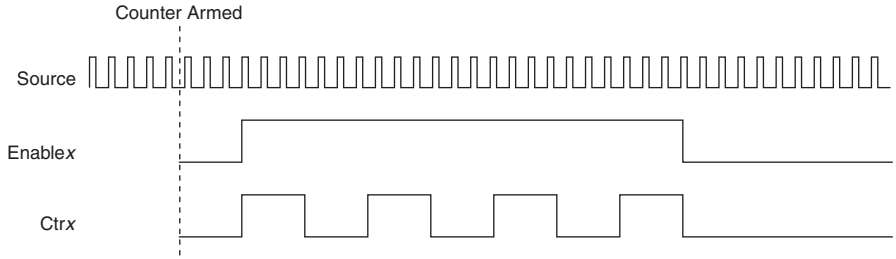
Refer to the following sections for more information about the pulse train generation options on the NI 6738/6739:

- *Finite Pulse Train Generation*
- *Retriggerable Pulse or Pulse Train Generation*
- *Continuous Pulse Train Generation*
- *Finite Implicit Buffered Pulse Train Generation*
- *Continuous Buffered Implicit Pulse Train Generation*
- *Finite Buffered Sample Clocked Pulse Train Generation*
- *Continuous Buffered Sample Clocked Pulse Train Generation*

Finite Pulse Train Generation

Finite pulse train generation creates a train of pulses with programmable frequency and duty cycle for a predetermined number of pulses, as shown in Figure 6-29. With NI 6738/6739 counters, the primary counter generates the specified pulse train and the embedded counter counts the pulses generated by the primary counter. When the embedded counter reaches the specified tick count, it generates a trigger that stops the primary counter generation.

Figure 6-29. Finite Pulse Train Generation: Four Ticks Initial Delay, Four Pulses

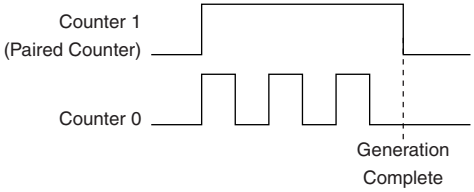


In Legacy Mode, the counter operation requires two counters and does not use the embedded counter. For example, to generate four pulses on Counter 0, Counter 0 generates the pulse train, which is gated by the paired second counter. The paired counter, Counter 1, generates a pulse of desired width.

Note Counter 0 is always paired with Counter 1. Counter 2 is always paired with Counter 3.

The routing is done internally. Figure 6-30 shows an example finite pulse train timing diagram.

Figure 6-30. Finite Pulse Train Timing in Legacy Mode



Retriggerable Pulse or Pulse Train Generation

The counter can output a single pulse or multiple pulses in response to each pulse on a hardware Start Trigger signal. The generated pulses appear on the Counter *n* Internal Output signal of the counter.

You can route the Start Trigger signal to the Gate input of the counter. You can specify a delay from the Start Trigger to the beginning of each pulse. You can also specify the pulse width. The delay and pulse width are measured in terms of a number of active edges of the Source input. The initial delay can be applied to only the first trigger or to all triggers using the CO.EnableInitalDelayOnRetrigger property. The default for a single pulse is True, while the default for finite pulse trains is False.

The counter ignores the Gate input while a pulse generation is in progress. After the pulse generation is finished, the counter waits for another Start Trigger signal to begin another pulse

generation. For retriggered pulse generation, pause triggers are not allowed since the pause trigger also uses the gate input.

Figure 6-31 shows a generation of two pulses with a pulse delay of five and a pulse width of three (using the rising edge of Source) with CO.EnableInitalDelayOnRetrigger set to the default True.

Figure 6-31. Retriggerable Single Pulse Generation with Initial Delay on Retrigger

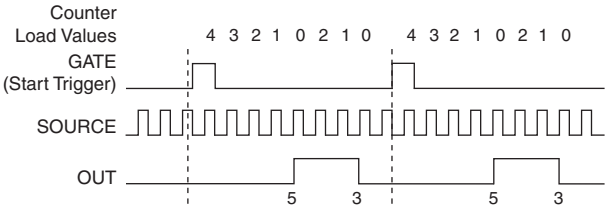
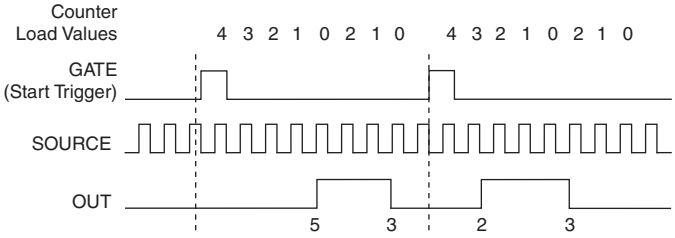


Figure 6-32 shows the same pulse train with CO.EnableInitalDelayOnRetrigger set to the default False.

Figure 6-32. Retriggerable Single Pulse Generation with Initial Delay on Retrigger Set to False



Note The minimum time between the trigger and the first active edge is two ticks of the source.

For information about connecting counter signals, refer to the *Default Counter/Timer Pinouts* section.

Continuous Pulse Train Generation

Continuous pulse train generation creates a train of pulses with programmable frequency and duty cycle. The pulses appear on the Counter *n* Internal Output signal of the counter.

You can specify a delay from when the counter is armed to the beginning of the pulse train. The delay is measured in terms of a number of active edges of the Source input.

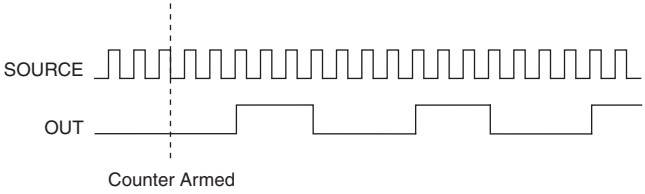
You specify the high and low pulse widths of the output signal. The pulse widths are also measured in terms of a number of active edges of the Source input. You can also specify the active edge of the Source input (rising or falling).

The counter can begin the pulse train generation as soon as the counter is armed, or in response to a hardware Start Trigger. You can route the Start Trigger to the Gate input of the counter.

You can also use the Gate input of the counter as a Pause Trigger (if it is not used as a Start Trigger). The counter pauses pulse generation when the Pause Trigger is active.

Figure 6-33 shows a continuous pulse train generation (using the rising edge of Source).

Figure 6-33. Continuous Pulse Train Generation



Continuous pulse train generation is sometimes called frequency division. If the high and low pulse widths of the output signal are *M* and *N* periods, then the frequency of the Counter *n* Internal Output signal is equal to the frequency of the Source input divided by *M* + *N*.

For information about connecting counter signals, refer to the *Default Counter/Timer Pinouts* section.

Buffered Pulse Train Generation

NI 6738/6739 counters can use the FIFO to perform a buffered pulse train generation. Buffered pulse train generation can use implicit timing or sample clock timing. When using implicit timing, the pulse idle time and active time changes with each sample you write. With sample clocked timing, each sample you write updates the idle time and active time of your generation on each sample clock edge. Idle time and active time can also be defined in terms of frequency and duty cycle or idle ticks and active ticks.

Note On buffered implicit pulse trains, the pulse specifications in the DAQmx Create Counter Output Channel are ignored so that you generate the number of pulses defined in the multipoint write. On buffered sample clock pulse trains, the pulse specifications in the DAQmx Create Counter Output Channel are generated after the counters start, and before the first sample clock, so that you generate the number of updates defined in the multipoint write.

Finite Implicit Buffered Pulse Train Generation

Finite implicit buffered pulse train generation creates a predetermined number of pulses with variable idle and active times. Each point you write generates a single pulse. The number of pairs

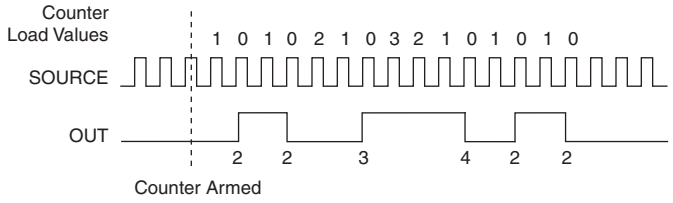
of idle and active times (pulse specifications) you write determines the number of pulses generated. All points are generated back to back to create a user defined pulse train.

Table 6-6 and Figure 6-34 detail a finite implicit generation of three samples.

Table 6-6. Finite Implicit Buffered Pulse Train Generation

Sample	Idle Ticks	Active Ticks
1	2	2
2	3	4
3	2	2

Figure 6-34. Finite Implicit Buffered Pulse Train Generation



Continuous Buffered Implicit Pulse Train Generation

Continuous buffered implicit pulse train generation creates a continuous train of pulses with variable idle and active times. Instead of generating a set number of data samples and stopping, a continuous generation continues until you stop the operation. Each point you write generates a single pulse. All points are generated back to back to create a user defined pulse train.

Finite Buffered Sample Clocked Pulse Train Generation

Finite buffered sample clocked pulse train generation creates a predetermined number of pulse train updates. Each point you write defines pulse specifications that are updated with each sample clock. When a sample clock occurs, the current pulse (idle followed by active) finishes generation and the next pulse updates with the next sample specifications.

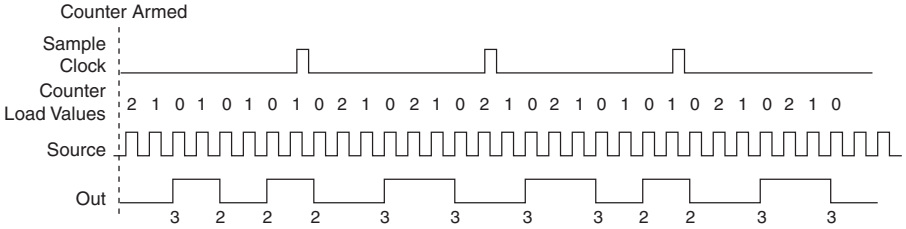
Note When the last sample is generated, the pulse train continues to generate with these specifications until the task is stopped.

Table 6-7 and Figure 6-35 detail a finite sample clocked generation of three samples where the pulse specifications from the create channel are two ticks idle, two ticks active, and three ticks initial delay.

Table 6-7. Finite Buffered Sample Clocked Pulse Train Generation

Sample	Idle Ticks	Active Ticks
1	3	3
2	2	2
3	3	3

Figure 6-35. Finite Buffered Sample Clocked Pulse Train Generation



There are several different methods of continuous generation that control what data is written. These methods are regeneration, FIFO regeneration, and non-regeneration modes.

Regeneration is the repetition of the data that is already in the buffer.

Standard regeneration is when data from the PC buffer is continually downloaded to the FIFO to be written out. New data can be written to the PC buffer at any time without disrupting the output. With FIFO regeneration, the entire buffer is downloaded to the FIFO and regenerated from there. Once the data is downloaded, new data cannot be written to the FIFO. To use FIFO regeneration, the entire buffer must fit within the FIFO size. The advantage of using FIFO regeneration is that it does not require communication with the main host memory once the operation is started, thereby preventing any problems that may occur due to excessive bus traffic.

With non-regeneration, old data is not repeated. New data must be continually written to the buffer. If the program does not write new data to the buffer at a fast enough rate to keep up with the generation, the buffer underflows and causes an error.

Continuous Buffered Sample Clocked Pulse Train Generation

Continuous buffered sample clocked pulse train generation creates a continuous train of pulses with variable idle and active times. Instead of generating a set number of data samples and stopping, a continuous generation continues until you stop the operation. Each point you write

specifies pulse specifications that are updated with each sample clock. When a sample clock occurs, the current pulse finishes generation and the next pulse uses the next sample specifications.

Frequency Generation

You can generate a frequency by using a counter in pulse train generation mode.

Frequency Division

The counters can generate a signal with a frequency that is a fraction of an input signal. This function is equivalent to continuous pulse train generation. Refer to the *Continuous Pulse Train Generation* section for detailed information.

For information about connecting counter signals, refer to the *Default Counter/Timer Pinouts* section.

Pulse Generation for ETS

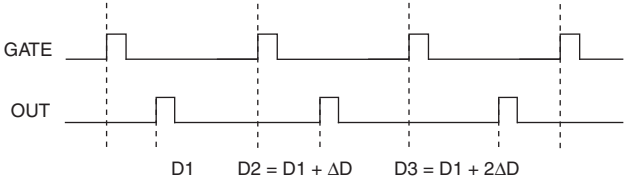
In the equivalent time sampling (ETS) application, the counter produces a pulse on the output a specified delay after an active edge on Gate. After each active edge on Gate, the counter cumulatively increments the delay between the Gate and the pulse on the output by a specified amount. Thus, the delay between the Gate and the pulse produced successively increases.

The increase in the delay value can be between 0 and 255. For instance, if you specify the increment to be 10, the delay between the active Gate edge and the pulse on the output increases by 10 every time a new pulse is generated.

Suppose you program your counter to generate pulses with a delay of 100 and pulse width of 200 each time it receives a trigger. Furthermore, suppose you specify the delay increment to be 10. On the first trigger, your pulse delay is 100, on the second it is 110, on the third it is 120; the process repeats until the counter is disarmed. The counter ignores any Gate edge that is received while the pulse triggered by the previous Gate edge is in progress.

The waveform thus produced at the counter's output can be used to provide timing for undersampling applications where a digitizing system can sample repetitive waveforms that are higher in frequency than the Nyquist frequency of the system. Figure 6-36 shows an example of pulse generation for ETS; the delay from the trigger to the pulse increases after each subsequent Gate active edge.

Figure 6-36. Pulse Generation for ETS



For information about connecting counter signals, refer to the *Default Counter/Timer Pinouts* section.

Counter Timing Signals

The NI 6738/6739 features the following counter timing signals:

- *Counter n Source Signal*
- *Counter n Gate Signal*
- *Counter n Aux Signal*
- *Counter n A Signal*
- *Counter n B Signal*
- *Counter n Z Signal*
- *Counter n Up_Down Signal*
- *Counter n HW Arm Signal*
- *Counter n Sample Clock Signal*
- *Counter n Internal Output Signal*
- *Counter n TC Signal*

Note All counter timing signals can be filtered. Refer to the *PFI Filters* section of Chapter 7, *PFI*, for more information.

In this section, *n* refers to the NI 6738/6739 Counter 0, 1, 2, or 3. For example, Counter *n* Source refers to four signals—Counter 0 Source (the source input to Counter 0), Counter 1 Source (the source input to Counter 1), Counter 2 Source (the source input to Counter 2), or Counter 3 Source (the source input to Counter 3).

Each of these signals supports digital filtering. Refer to the *PFI Filters* section of Chapter 7, *PFI*, for more information.

Counter *n* Source Signal

The selected edge of the Counter *n* Source signal increments and decrements the counter value depending on the application the counter is performing. Table 6-8 lists how the terminal is used in various applications.

Table 6-8. Counter Applications and Counter *n* Source

Application	Purpose of Source Terminal
Pulse Generation	Counter Timebase
One Counter Time Measurements	Counter Timebase

Table 6-8. Counter Applications and Counter *n* Source

Application	Purpose of Source Terminal
Two Counter Time Measurements	Input Terminal
Non-Buffered Edge Counting	Input Terminal
Buffered Edge Counting	Input Terminal
Two-Edge Separation	Counter Timebase

Routing a Signal to Counter *n* Source

Each counter has independent input selectors for the Counter *n* Source signal. Any of the following signals can be routed to the Counter *n* Source input:

- 100 MHz Timebase
- 20 MHz Timebase
- 100 kHz Timebase
- RTSI <0..7>
- PXI_Trig<0..7>
- (NI PCIe/PXIe-6738) PFI <0..7>; (NI PXIe-6739) PFI <0..15>
- PXI_CLK10
- PXI_STAR
- PXIe_DSTAR<A,B>
- Change Detection Event

In addition, TC or Gate from a counter can be routed to a different counter source.

Some of these options may not be available in some driver software.

Routing Counter *n* Source to an Output Terminal

You can route Counter *n* Source out to any PFI <0..7>/<8..15>, RTSI <0..7>, PXI_Trig<0..7>, or PXIe_DSTARC terminal. All PFIs are set to high-impedance at startup.

Counter *n* Gate Signal

The Counter *n* Gate signal can perform many different operations depending on the application including starting and stopping the counter, and saving the counter contents.

Routing a Signal to Counter *n* Gate

Each counter has independent input selectors for the Counter *n* Gate signal. Any of the following signals can be routed to the Counter *n* Gate input:

- PXI_Trig<0..7>
- (NI PCIe/PXIe-6738) PFI <0..7>; (NI PXIe-6739) PFI <0..15>

- RTSI <0..7>
- DI Sample Clock (di/SampleClock)
- DI Reference Trigger (di/ReferenceTrigger)
- DO Sample Clock (do/SampleClock)
- PXI_STAR
- PXIe_DSTAR<A,B>
- Change Detection Event

In addition, a counter's Internal Output or Source can be routed to a different counter's gate.

Some of these options may not be available in some driver software.

Routing Counter *n* Gate to an Output Terminal

You can route Counter *n* Gate out to any PFI <0..7>/<8..15>, RTSI <0..7>, PXI_Trig<0..7>, or PXIe_DSTARC terminal. All PFIs are set to high-impedance at startup.

Counter *n* Aux Signal

The Counter *n* Aux signal indicates the first edge in a two-signal edge-separation measurement.

Routing a Signal to Counter *n* Aux

Each counter has independent input selectors for the Counter *n* Aux signal. Any of the following signals can be routed to the Counter *n* Aux input:

- PXI_Trig<0..7>
- (NI PCIe/PXIe-6738) PFI <0..7>; (NI PXIe-6739) PFI <0..15>
- RTSI <0..7>
- PXI_STAR
- PXIe_DSTAR<A,B>
- Change Detection Event

In addition, a counter's Internal Output, Gate or Source can be routed to a different counter's Aux. A counter's own gate can also be routed to its Aux input.

Some of these options may not be available in some driver software.

Counter *n* A, Counter *n* B, and Counter *n* Z Signals

Counter *n* B can control the direction of counting in edge counting applications. Use the A, B, and Z inputs to each counter when measuring quadrature encoders or measuring two pulse encoders.

Routing Signals to A, B, and Z Counter Inputs

Each counter has independent input selectors for each of the A, B, and Z inputs. Any of the following signals can be routed to each input:

- PXI_Trig<0..7>
- (NI PCIe/PXIe-6738) PFI <0..7>; (NI PXIe-6739) PFI <0..15>
- RTSI <0..7>
- PXI_STAR
- PXIe_DSTAR<A,B>

Routing Counter *n* Z Signal to an Output Terminal

You can route Counter *n* Z out to any RTSI <0..7> or PXI_Trig<0..7> terminal.

Counter *n* Up_Down Signal

Counter *n* Up_Down is another name for the Counter *n* B signal.

Counter *n* HW Arm Signal

The Counter *n* HW Arm signal enables a counter to begin an input or output function.

To begin any counter input or output function, you must first enable, or arm, the counter. In some applications, such as a buffered edge count, the counter begins counting when it is armed. In other applications, such as single pulse-width measurement, the counter begins waiting for the Gate signal when it is armed. Counter output operations can use the arm signal in addition to a start trigger.

Software can arm a counter or configure counters to be armed on a hardware signal. Software calls this hardware signal the Arm Start Trigger. Internally, software routes the Arm Start Trigger to the Counter *n* HW Arm input of the counter.

Routing Signals to Counter *n* HW Arm Input

Any of the following signals can be routed to the Counter *n* HW Arm input:

- PXI_Trig<0..7>
- (NI PCIe/PXIe-6738) PFI <0..7>; (NI PXIe-6739) PFI <0..15>
- RTSI <0..7>
- PXI_STAR
- PXIe_DSTAR<A,B>
- Change Detection Event

A counter's Internal Output can be routed to a different counter's HW Arm.

Some of these options may not be available in some driver software.

Counter *n* Sample Clock Signal

Use the Counter *n* Sample Clock (Ctr*n*SampleClock) signal to perform sample clocked acquisitions and generations.

You can specify an internal or external source for Counter *n* Sample Clock. You can also specify whether the measurement sample begins on the rising edge or falling edge of Counter *n* Sample Clock.

If the DAQ device receives a Counter *n* Sample Clock when the FIFO is full, it reports an overflow error to the host software.

Using an Internal Source

To use Counter *n* Sample Clock with an internal source, specify the signal source and the polarity of the signal. The source can be any of the following signals:

- DI Sample Clock (di/SampleClock)
- DO Sample Clock (do/SampleClock)
- DI Change Detection output
- AO Sample Clock

Several other internal signals can be routed to Counter *n* Sample Clock through internal routes. Refer to *Device Routing in MAX* in the *NI-DAQmx Help* or the *LabVIEW Help* for more information.

Using an External Source

You can route any of the following signals as Counter *n* Sample Clock:

- (NI PCIe/PXIe-6738) PFI <0..7>; (NI PXIe-6739) PFI <0..15>
- RTSI <0..7>
- PXI_Trig<0..7>
- PXI_STAR
- PXIe_DSTAR<A,B>

You can sample data on the rising or falling edge of Counter *n* Sample Clock.

Routing Counter *n* Sample Clock to an Output Terminal

You can route Counter *n* Sample Clock out to any PFI <0..7>/<8..15> terminal. The PFI circuitry inverts the polarity of Counter *n* Sample Clock before driving the PFI terminal.

Counter *n* Internal Output and Counter *n* TC Signals

The Counter *n* Internal Output signal changes in response to Counter *n* TC.

The two software-selectable output options are pulse output on TC and toggle output on TC. The output polarity is software-selectable for both options.

With pulse or pulse train generation tasks, the counter drives the pulse(s) on the Counter *n* Internal Output signal. The Counter *n* Internal Output signal can be internally routed to be a counter/timer input or an “external” source for AI, AO, DI, or DO timing signals.

Routing Counter *n* Internal Output to an Output Terminal

You can route Counter *n* Internal Output to any PFI <0..7>/<8..15>, RTSI <0..7>, or PXIe_DSTARC terminal. All PFIs are set to high-impedance at startup.

Default Counter/Timer Pinouts

By default, NI-DAQmx routes the counter/timer inputs and outputs to the PFI pins.

Refer to Table 6-9 for the default NI-DAQmx counter/timer outputs for the NI PCIe/PXIe-6738. Refer to Table 6-10 for the default NI-DAQmx counter/timer outputs for the NI PXIe-6739.

Table 6-9. NI PCIe/PXIe-6738 Default NI-DAQmx Counter/Timer Pins

Counter/Timer Signal	Connector 0 Pin Number (Name)
CTR 0 SRC	7 (PFI 5)
CTR 0 GATE	8 (PFI 6)
CTR 0 AUX	40 (PFI 4)
CTR 0 OUT	9 (PFI 7)
CTR 0 A	7 (PFI 5)
CTR 0 Z	8 (PFI 6)
CTR 0 B	40 (PFI 4)
CTR 1 SRC	4 (PFI 0)
CTR 1 GATE	38 (PFI 1)
CTR 1 AUX	6 (PFI 3)
CTR 1 OUT	5 (PFI 2)
CTR 1 A	4 (PFI 0)
CTR 1 Z	38 (PFI 1)
CTR 1 B	6 (PFI 3)
CTR 2 SRC	7 (PFI 5)
CTR 2 GATE	8 (PFI 6)
CTR 2 AUX	40 (PFI 4)

Table 6-9. NI PCIe/PXIe-6738 Default NI-DAQmx Counter/Timer Pins (Continued)

Counter/Timer Signal	Connector 0 Pin Number (Name)
CTR 2 OUT	9 (PFI 7)
CTR 2 A	7 (PFI 5)
CTR 2 Z	8 (PFI 6)
CTR 2 B	40 (PFI 4)
CTR 3 SRC	4 (PFI 0)
CTR 3 GATE	38 (PFI 1)
CTR 3 AUX	6 (PFI 3)
CTR 3 OUT	5 (PFI 2)
CTR 3 A	4 (PFI 0)
CTR 3 Z	38 (PFI 1)
CTR 3 B	6 (PFI 3)

Table 6-10. NI PXIe-6739 Default NI-DAQmx Counter/Timer Pins

Counter/Timer Signal	Connector 0 Pin Number (Name)	Connector 1 Pin Number (Name)
CTR 0 SRC	7 (PFI 5)	—
CTR 0 GATE	8 (PFI 6)	—
CTR 0 AUX	40 (PFI 4)	—
CTR 0 OUT	9 (PFI 7)	—
CTR 0 A	7 (PFI 5)	—
CTR 0 Z	8 (PFI 6)	—
CTR 0 B	40 (PFI 4)	—
CTR 1 SRC	4 (PFI 0)	—
CTR 1 GATE	38 (PFI 1)	—
CTR 1 AUX	6 (PFI 3)	—
CTR 1 OUT	5 (PFI 2)	—
CTR 1 A	4 (PFI 0)	—

Table 6-10. NI PXIe-6739 Default NI-DAQmx Counter/Timer Pins (Continued)

Counter/Timer Signal	Connector 0 Pin Number (Name)	Connector 1 Pin Number (Name)
CTR 1 Z	38 (PFI 1)	—
CTR 1 B	6 (PFI 3)	—
CTR 2 SRC	—	7 (PFI 13)
CTR 2 GATE	—	8 (PFI 14)
CTR 2 AUX	—	40 (PFI 12)
CTR 2 OUT	—	9 (PFI 15)
CTR 2 A	—	7 (PFI 13)
CTR 2 Z	—	8 (PFI 14)
CTR 2 B	—	40 (PFI 12)
CTR 3 SRC	—	4 (PFI 8)
CTR 3 GATE	—	38 (PFI 9)
CTR 3 AUX	—	6 (PFI 11)
CTR 3 OUT	—	5 (PFI 10)
CTR 3 A	—	4 (PFI 8)
CTR 3 Z	—	38 (PFI 9)
CTR 3 B	—	6 (PFI 11)

You can use these defaults or select other sources and destinations for the counter/timer signals in NI-DAQmx. Refer to *Connecting Counter Signals* in the *NI-DAQmx Help* or the *LabVIEW Help* for more information about how to connect your signals for common counter measurements and generations. NI 6738/6739 default PFI lines for counter functions are listed in *NI 6738/6739 Physical Channels* in the *NI-DAQmx Help* or the *LabVIEW Help*.

Counter Triggering

Counters support three different triggering actions:

- **Arm Start Trigger**—To begin any counter input or output function, you must first enable, or arm, the counter. Software can arm a counter or configure counters to be armed on a hardware signal. Software calls this hardware signal the Arm Start Trigger. Internally, software routes the Arm Start Trigger to the Counter *n* HW Arm input of the counter.

  For counter output operations, you can use it in addition to the start and pause triggers. For counter input operations, you can use the arm start trigger to have start trigger-like behavior. The arm start trigger can be used for synchronizing multiple counter input and output tasks.

  When using an arm start trigger, the arm start trigger source is routed to the Counter *n* HW Arm signal.
- **Start Trigger**—For counter output operations, a start trigger can be configured to begin a finite or continuous pulse generation. Once a continuous generation has triggered, the pulses continue to generate until you stop the operation in software. For finite generations, the specified number of pulses is generated and the generation stops unless you use the retriggerable attribute. When you use this attribute, subsequent start triggers cause the generation to restart.

  When using a start trigger, the start trigger source is routed to the Counter *n* Gate signal input of the counter.

  Counter input operations can use the arm start trigger to have start trigger-like behavior.
- **Pause Trigger**—You can use pause triggers in edge counting and continuous pulse generation applications. For edge counting acquisitions, the counter stops counting edges while the external trigger signal is low and resumes when the signal goes high or vice versa. For continuous pulse generations, the counter stops generating pulses while the external trigger signal is low and resumes when the signal goes high or vice versa.

  When using a pause trigger, the pause trigger source is routed to the Counter *n* Gate signal input of the counter.

Other Counter Features

The following sections list the other counter features available on the NI 6738/6739.

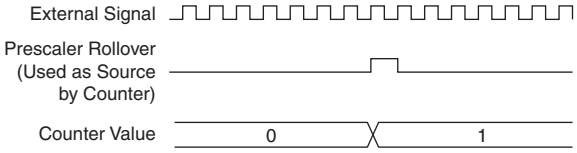
Cascading Counters

You can internally route the Counter *n* Internal Output and Counter *n* TC signals of each counter to the Gate inputs of the other counter. By cascading two counters together, you can effectively create a 64-bit counter. By cascading counters, you can also enable other applications. For example, to improve the accuracy of frequency measurements, use reciprocal frequency measurement, as described in the *Large Range of Frequencies with Two Counters* section.

Prescaling

Prescaling allows the counter to count a signal that is faster than the maximum timebase of the counter, as shown in Figure 6-37. The NI 6738/6739 offers 8X and 2X prescaling on each counter (prescaling can be disabled). Each prescaler consists of a small, simple counter that counts to eight (or two) and rolls over. This counter can run faster than the larger counters, which simply count the rollovers of this smaller counter. Thus, the prescaler acts as a frequency divider on the Source and puts out a frequency that is one-eighth (or one-half) of what it is accepting.

Figure 6-37. Prescaling



Prescaling is intended to be used for frequency measurement where the measurement is made on a continuous, repetitive signal. The prescaling counter cannot be read; therefore, you cannot determine how many edges have occurred since the previous rollover. Prescaling can be used for event counting provided it is acceptable to have an error of up to seven (or one) ticks. Prescaling can be used when the counter Source is an external signal. Prescaling is not available if the counter Source is one of the internal timebases (100MHzTimebase, 20MHzTimebase, or 100kHzTimebase).

Synchronization Modes

The 32-bit counter counts up or down synchronously with the Source signal. The Gate signal and other counter inputs are asynchronous to the Source signal, so the NI 6738/6739 synchronizes these signals before presenting them to the internal counter.

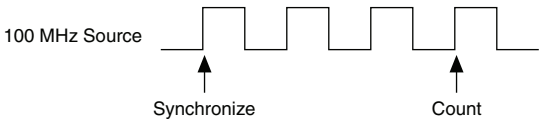
Depending on how you configure your device, the NI 6738/6739 uses one of three synchronization methods:

- *100 MHz Source Mode*
- *External Source Greater than 25 MHz*
- *External or Internal Source Less than 25 MHz*

100 MHz Source Mode

In 100 MHz source mode, the device synchronizes signals on the rising edge of the source, and counts on the third rising edge of the source. Edges are pipelined so no counts are lost, as shown in Figure 6-38.

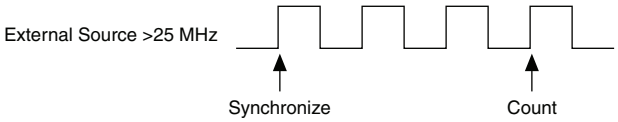
Figure 6-38. 100 MHz Source Mode



External Source Greater than 25 MHz

With an external source greater than 25 MHz, the device synchronizes signals on the rising edge of the source, and counts on the third rising edge of the source. Edges are pipelined so no counts are lost, as shown in Figure 6-39.

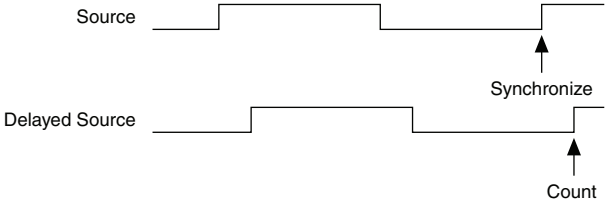
Figure 6-39. External Source Greater than 25 MHz



External or Internal Source Less than 25 MHz

With an external or internal source less than 25 MHz, the device generates a delayed Source signal by delaying the Source signal by several nanoseconds. The device synchronizes signals on the rising edge of the delayed Source signal, and counts on the following rising edge of the source, as shown in Figure 6-40.

Figure 6-40. External or Internal Source Less than 25 MHz



7

PFI

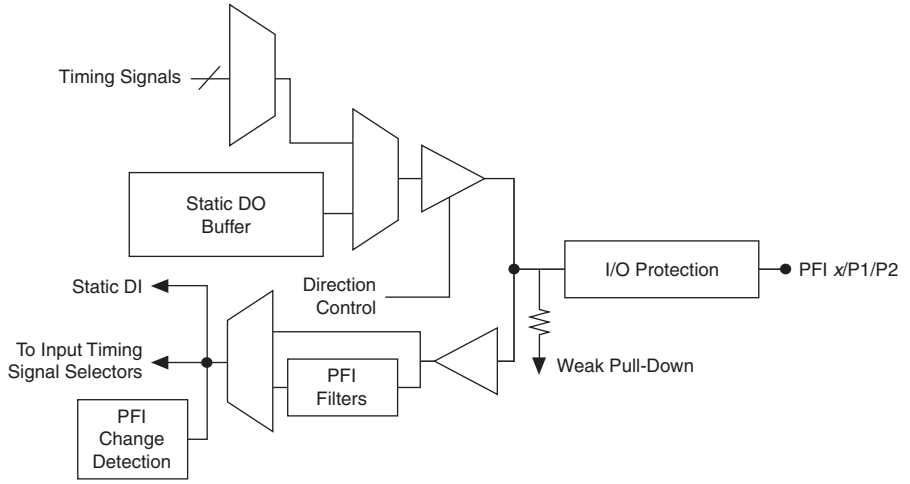
The NI 6738/6739 has up to 16 Programmable Function Interface (PFI) signals. In addition, the NI 6738/6739 has up to four lines of bidirectional DIO signals.

Each PFI can be individually configured as the following:

- A static digital input
- A static digital output
- A timing input signal for AO, DI, DO, or counter/timer functions
- A timing output signal from AO, DI, DO, or counter/timer functions

Each PFI input also has a programmable debouncing filter. Figure 7-1 shows the circuitry of one PFI line. Each PFI line is similar.

Figure 7-1. NI 6738/6739 PFI Circuitry



When a terminal is used as a timing input or output signal, it is called PFI *x* (where *x* is an integer from 0 to 15). When a terminal is used as a static digital input or output, it is called P1.*x* or P2.*x*. On the I/O connector, each terminal is labeled PFI *x*/P1.*x* or PFI *x*/P2.*x*.

The voltage input and output levels and the current drive levels of the PFI signals are listed in the specifications of your device.

Using PFI Terminals as Timing Input Signals

Use PFI terminals to route external timing signals to many different device functions. Each PFI terminal can be routed to any of the following signals:

- Counter input signals for all counters—Source, Gate, Aux, HW_Arm, A, B, Z
- Counter *n* Sample Clock
- DI Sample Clock (di/SampleClock)
- DI Sample Clock Timebase (di/SampleClockTimebase)
- DI Reference Trigger (di/ReferenceTrigger)
- DI Start Trigger
- DI Pause Trigger
- DO Sample Clock (do/SampleClock)
- DO Start Trigger
- DO Pause Trigger
- AO Sample Clock
- AO Start Trigger
- AO Pause Trigger

Most functions allow you to configure the polarity of PFI inputs and whether the input is edge or level sensitive.

Exporting Timing Output Signals Using PFI Terminals

You can route any of the following timing signals to any PFI terminal configured as an output:

- DI Sample Clock (di/SampleClock)
- DI Start Trigger (di/StartTrigger)
- DI Reference Trigger (di/ReferenceTrigger)
- DI Pause Trigger (di/PauseTrigger)
- DO Sample Clock* (do/SampleClock)
- DO Start Trigger (do/StartTrigger)
- DO Pause Trigger (do/PauseTrigger)
- AO Sample Clock
- AO Start Trigger
- AO Pause Trigger
- Counter *n* Source
- Counter *n* Gate

- Counter *n* Internal Output
- Counter *n* Sample Clock
- Counter *n* Counter *n* HW Arm
- PXI_STAR
- RTSI <0..7>
- PXI_Trig<0..7>
- Change Detection Event
- Watchdog timer expired pulse

Note Signals with an * are inverted before being driven to a terminal; that is, these signals are active low.

Using PFI Terminals as Static Digital I/Os

Each PFI can be individually configured as a static digital input or a static digital output. When a terminal is used as a static digital input or output, it is called P1.*x* or P2.*x*. On the I/O connector, each terminal is labeled PFI *x*/P1.*x* or PFI *x*/P2.*x*.

In addition, the NI 6738/6739 has up to 20 lines of bidirectional DIO signals. For more information on the digital I/O functionality, refer to Chapter 5, *Digital I/O*.

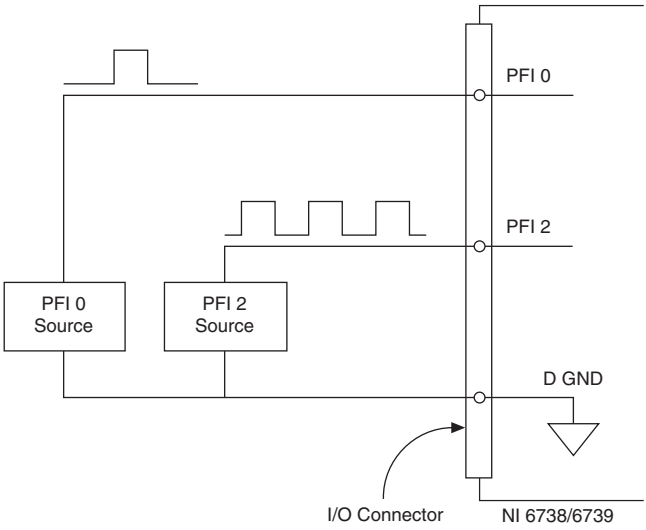
Using PFI Terminals to Digital Detection Events

Each PFI can be configured to detect digital changes. The values on the PFI lines cannot be read in a hardware-timed task, but they can be used to fire the change detection event. For example, if you wanted to do change detection on eight timed DIO lines but wanted to ensure that the value of the lines was updated every second independent of the eight lines changing you could set a PFI line up for change detection and connect a 1 Hz signal to it. For more information on the digital change detection, refer to the *DI Change Detection* section of Chapter 5, *Digital I/O*.

Connecting PFI Input Signals

All PFI input connections are referenced to D GND. Figure 7-2 shows this reference, and how to connect an external PFI 0 source and an external PFI 2 source to two PFI terminals.

Figure 7-2. PFI Input Signal Connections



PFI Filters

You can enable a programmable debouncing filter on each PFI, RTSI, PXI_Trig, PXI_STAR, or PXIe_DSTAR<A,B> signal. When the filters are enabled, your device samples the input on each rising edge of a filter clock. The NI 6738/6739 uses an onboard oscillator to generate the filter clock.

The following is an example of low to high transitions of the input signal. High-to-low transitions work similarly.

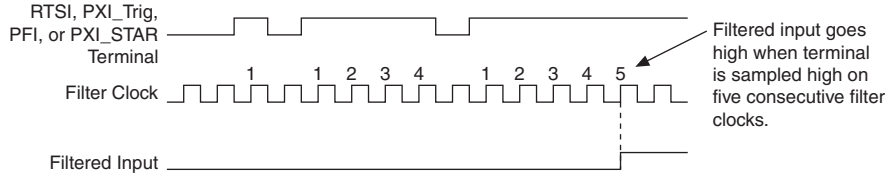
Assume that an input terminal has been low for a long time. The input terminal then changes from low to high, but glitches several times. When the filter clock has sampled the signal high on N consecutive edges, the low to high transition is propagated to the rest of the circuit. The value of N depends on the filter setting; refer to Table 7-1.

Table 7-1. Filters

Filter Setting	Filter Clock	N (Filter Clocks Needed to Pass Signal)	Pulse Width Guaranteed to Pass Filter	Pulse Width Guaranteed to Not Pass Filter
None	—	—	—	—
90 ns (short)	100 MHz	9	90 ns	80 ns
5.12 µs (medium)	100 MHz	512	5.12 µs	5.11 µs
2.56 ms (high)	100 kHz	256	2.56 ms	2.55 ms
Custom	User configurable	N	N/timebase	(N - 1)/ timebase

The filter setting for each input can be configured independently. On power up, the filters are disabled. Figure 7-3 shows an example of a low to high transition on an input that has a custom filter set to N = 5.

Figure 7-3. Filter Example



Enabling filters introduces jitter on the input signal. The maximum jitter is one period of the timebase.

When a PXI_Trig or RTSI input is routed directly to PFI, the device does not use the filtered version of the input signal.

I/O Protection

Each DIO and PFI signal is protected against overvoltage, undervoltage, and overcurrent conditions as well as ESD events. However, you should avoid these fault conditions by following these guidelines:

- If you configure a PFI or DIO line as an output, do *not* connect it to any external signal source, ground, or power supply.
- If you configure a PFI or DIO line as an output, understand the current requirements of the load connected to these signals. Do *not* exceed the specified current output limits of the DAQ device. NI has several signal conditioning solutions for digital applications requiring high current drive.
- If you configure a PFI or DIO line as an input, do *not* drive the line with voltages outside of its normal operating range.
- Treat the DAQ device as you would treat any static sensitive device. *Always* properly ground yourself and the equipment when handling the DAQ device or connecting to it.

Programmable Power-Up States

At system startup and reset, the hardware sets all PFI and DIO lines to high-impedance inputs by default. The DAQ device does not drive the signal high or low. Each line has a weak pull-down resistor connected to it, as described in the specifications document for your device.

NI-DAQmx supports programmable power-up states for PFI and DIO lines. Software can program any value at power up to the P0, P1, or P2 lines. The PFI and DIO lines can be set as:

- A high-impedance input with a weak pull-down resistor (default)
- An output driving a 0
- An output driving a 1

Refer to the *NI-DAQmx Help* or the *LabVIEW Help* for more information about setting power-up states in NI-DAQmx or MAX.

8

Digital Routing and Clock Generation

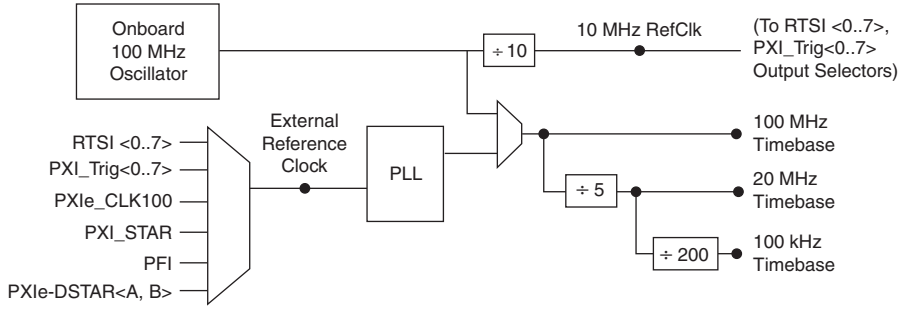
The digital routing circuitry has the following main functions:

- Routes timing and control signals. The acquisition/generation sub-systems use these signals to manage acquisitions and generations. These signals can come from the following sources:
  - Your NI 6738/6739 device
  - Other devices in your system through PXI_Trig or RTSI
  - User input through the PFI terminals
  - User input through the PXI_STAR terminal
- Routes and generates the main clock signals for the NI 6738/6739.

Clock Routing

Figure 8-1 shows the clock routing circuitry of the NI 6738/6739.

Figure 8-1. NI 6738/6739 Clock Routing Circuitry



100 MHz Timebase

The 100 MHz Timebase can be used as the timebase for all internal subsystems.

The 100 MHz Timebase is generated from the following sources:

- Onboard oscillator
- External signal (by using the external reference clock)

20 MHz Timebase

The 20 MHz Timebase can be used to generate many of the AO timing signals.The 20 MHz Timebase can also be used as the Source input to the 32-bit general-purpose counter/timers.

The 20 MHz Timebase is generated by dividing down the 100 MHz Timebase.

100 kHz Timebase

The 100 kHz Timebase can be used to generate many of the AO timing signals. The 100 kHz Timebase can also be used as the Source input to the 32-bit general-purpose counter/timers.

The 100 kHz Timebase is generated by dividing down the 20 MHz Timebase by 200.

External Reference Clock

The external reference clock can be used as a source for the internal timebases (100 MHz Timebase, 20 MHz Timebase, and 100 kHz Timebase) on an NI 6738/6739. By using the external reference clock, you can synchronize the internal timebases to an external clock.

The following signals can be routed to drive the external reference clock:

- PXI_Trig<0..7>
- (NI PCIe/PXIe-6738) PFI <0..7>; (NI PXIe-6739) PFI <0..15>
- RTSI <0..7>
- PXIe_CLK100
- PXI_STAR
- PXIe_DSTAR<A,B>

The external reference clock is an input to a Phase-Lock Loop (PLL). The PLL generates the internal timebases.

Caution Do *not* disconnect an external reference clock once the devices have been synchronized or are used by a task. Doing so may cause the device to go into an unknown state. Make sure that all tasks using a reference clock are stopped before disconnecting it.

Enabling or disabling the PLL through the use of a reference clock affects the clock distribution to all subsystems. For this reason, the PLL can only be enabled or disabled when no other tasks are running in any of the device subsystems.

10 MHz Reference Clock

The 10 MHz reference clock can be used to synchronize other devices to your device. The 10 MHz reference clock can be routed to the RTSI <0..7>, PXI_Trig<0..7>, or PFI <0..7>/<8..15> terminals.

The 10 MHz reference clock is generated by dividing down the onboard oscillator.

Synchronizing Multiple Devices

The following sections contain information about synchronizing multiple NI 6738/6739 devices.

PXI Express Devices

On PXI Express systems, you can synchronize devices to PXIe_CLK100. In this application, the PXI Express chassis acts as the initiator. Each PXI Express module routes PXIe_CLK100 to its external reference clock.

Another option in PXI Express systems is to use PXI_STAR. The Star Trigger controller device acts as the initiator and drives PXI_STAR with a clock signal. Each target device routes PXI_STAR to its external reference clock.

PCI Express Devices

With the RTSI and PFI buses and the routing capabilities of PCI Express devices, there are several ways to synchronize multiple devices depending on your application.

To synchronize multiple devices to a common timebase, choose one device—the initiator—to generate the timebase. The initiator device routes its 10 MHz reference clock to one of the RTSI <0..7> or PFI <0..7> signals.

All devices (including the initiator device) receive the 10 MHz reference clock from RTSI or PFI. This signal becomes the external reference clock. A PLL on each device generates the internal timebases synchronous to the external reference clock.

Once all of the devices are using or referencing a common timebase, you can synchronize operations across them by sending a common start trigger out across the RTSI or PFI bus and setting their sample clock rates to the same value.

Real-Time System Integration (RTSI)

Real-Time System Integration (RTSI) is a signal bus among devices that allows you to do the following:

- Use a common clock (or timebase) to drive the timing engine on multiple devices
- Share trigger signals between devices

Many National Instruments DAQ, motion, vision, and CAN devices support RTSI.

In a PCI Express system, the RTSI bus consists of the RTSI bus interface and a ribbon cable. The bus can route timing and trigger signals between several functions on as many as five DAQ, vision, motion, or CAN devices in the computer.

In a PXI Express system, the RTSI bus is replaced by the PXI and PXI Express trigger signals on the PXI Express backplane. This bus can route timing and trigger signals between several functions for all DAQ devices installed in a single PXI Express chassis. By using a PXI Synchronization Module, triggers can be shared across multiple PXI Express chassis.

RTSI Connector Pinout

(NI PCIe-6738 Devices) Figure 8-2 shows the RTSI connector pinout and Table 8-1 describes the RTSI signals.

Figure 8-2. PCI Express X Series Device RTSI Pinout

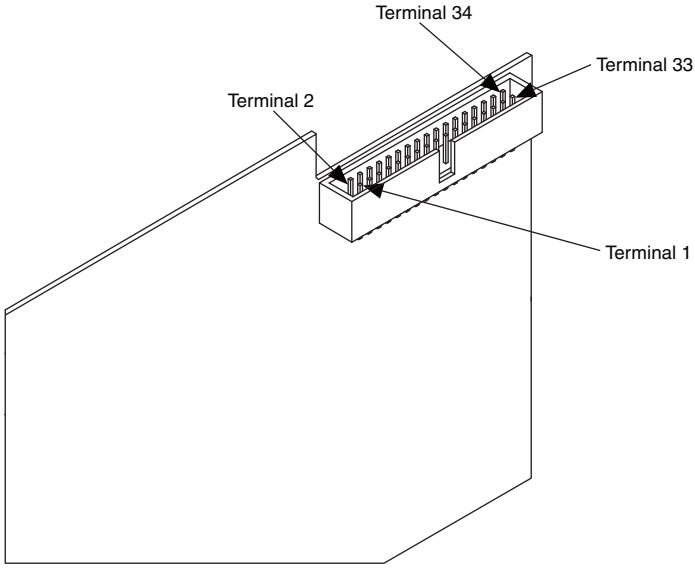


Table 8-1. RTSI Signals

RTSI Bus Signal	Terminal
RTSI 7	34
RTSI 6	32
RTSI 5	30
RTSI 4	28
RTSI 3	26
RTSI 2	24
RTSI 1	22

Table 8-1. RTSI Signals (Continued)

RTSI Bus Signal	Terminal
RTSI 0	20
Not Connected. Do not connect signals to these terminals.	1 through 18
D GND	19, 21, 23, 25, 27, 29, 31, 33

Using RTSI as Outputs

RTSI <0..7> are bidirectional terminals. As an output, you can drive any of the following signals to any RTSI terminal:

- AO Sample Clock* (ao/SampleClock)
- AO Start Trigger (ao/StartTrigger)
- AO Pause Trigger (ao/PauseTrigger)
- DI Start Trigger (di/StartTrigger)
- DI Sample Clock (di/SampleClock)
- DI Pause Trigger (di/PauseTrigger)
- DI Reference Trigger (di/ReferenceTrigger)
- DO Start Trigger (do/StartTrigger)
- DO Sample Clock* (do/SampleClock)
- DO Pause Trigger (do/PauseTrigger)
- 10 MHz Reference Clock
- Counter *n* Source, Gate, Z, Internal Output
- Change Detection Event
- Analog Comparison Event
- (NI PCIe/PXIe-6738) PFI <0..7>; (NI PXIe-6739) PFI <0..15>

Note Signals with an * are inverted before being driven on the RTSI terminals.

Using RTSI Terminals as Timing Input Signals

You can use RTSI terminals to route external timing signals to many different functions. Each RTSI terminal can be routed to any of the following signals:

- AO Start Trigger (ao/StartTrigger)
- AO Sample Clock (ao/SampleClock)
- AO Sample Clock Timebase (ao/SampleClockTimebase)
- AO Pause Trigger (ao/PauseTrigger)
- Counter input signals for all counters—Source, Gate, Aux, HW_Arm, A, B, or Z

- DI Sample Clock (di/SampleClock)
- DI Start Trigger (di/StartTrigger)
- DI Pause Trigger (di/PauseTrigger)
- DI Reference Trigger (di/ReferenceTrigger)
- DO Sample Clock (do/SampleClock)
- DO Sample Clock Timebase (do/SampleClockTimebase)

Most functions allow you to configure the polarity of RTSI inputs and whether the input is edge or level sensitive.

RTSI Filters

You can enable a programmable debouncing filter on each PFI, RTSI, PXI_Trig, or PXI_STAR signal. Refer to the *PFI Filters* section of Chapter 7, *PFI*, for more information.

PXI and PXI Express Clock and Trigger Signals

PXIe_CLK100

PXIe_CLK100 is a common low-skew 100 MHz reference clock for synchronization of multiple modules in a PXI Express measurement or control system. The PXIe backplane is responsible for generating PXIe_CLK100 independently to each peripheral slot in a PXI Express chassis. For more information, refer to the *PXI Express Specification* at `www.pxisa.org`.

PXIe_SYNC100

PXIe_SYNC100 is a common low-skew 10 MHz reference clock with a 10% duty cycle for synchronization of multiple modules in a PXI Express measurement or control system. This signal is used to accurately synchronize modules using PXIe_CLK100 along with those using PXI_CLK10. The PXI Express backplane is responsible for generating PXIe_SYNC100 independently to each peripheral slot in a PXI Express chassis. For more information, refer to the *PXI Express Specification* at `www.pxisa.org`.

PXI_CLK10

PXI_CLK10 is a common low-skew 10 MHz reference clock for synchronization of multiple modules in a PXI measurement or control system. The PXI backplane is responsible for generating PXI_CLK10 independently to each peripheral slot in a PXI chassis.

Note PXI_CLK10 cannot be used as a reference clock for the NI PXIe-6738/6739.

PXI Triggers

A PXI chassis provides eight bused trigger lines to each module in a system. Triggers may be passed from one module to another, allowing precisely timed responses to asynchronous external events that are being monitored or controlled. Triggers can be used to synchronize the operation of several different PXI peripheral modules.

Note that in a PXI chassis with more than eight slots, the PXI trigger lines may be divided into multiple independent buses. Refer to the documentation for your chassis for details.

PXI_STAR Trigger

In a PXI Express system, the Star Trigger bus implements a dedicated trigger line between the system timing slot and the other peripheral slots. The Star Trigger can be used to synchronize multiple devices or to share a common trigger signal among devices.

A PXI Synchronization Module can be installed in the system timing slot to provide trigger signals to other peripheral modules. Systems that do not require this functionality can install any standard peripheral module in this system timing slot.

The NI PXIe-6738/6739 receives the Star Trigger signal (PXI_STAR) from a PXI Synchronization Module. PXI_STAR can be used as an external source for many AO and counter signals.

The NI PXIe-6738/6739 cannot source PXI_Star to other modules. The NI PXIe-6738/6739 can be used in the system timing slot of a PXI system, but the system will not be able to use the Star Trigger feature.

PXI_STAR Filters

You can enable a programmable debouncing filter on each PFI, PXIe_DSTAR, PXI_Trig, or PXI_STAR signal. Refer to the *PFI Filters* section of Chapter 7, *PFI*, for more information.

PXIe_DSTAR<A..C>

PXI Express devices can provide high-quality and high-frequency point-to-point connections between each slot and a system timing slot. These connections come in the form of three low-voltage differential star triggers that create point-to-point, high-frequency connections between a PXI Synchronization Module and a peripheral device. Using multiple connections enable you to create more applications because of the increased routing capabilities.

Table 8-2 describes the three differential star (DSTAR) lines and how they are used.

Table 8-2. PXIe_DSTAR Line Descriptions

Trigger Line	Purpose
PXIe_DSTARA	Distributes high-speed, high-quality clock signals from the system timing slot to the peripherals (input).
PXIe_DSTARB	Distributes high-speed, high-quality trigger signals from the system timing slot to the peripherals (input).
PXIe_DSTARC	Sends high-speed, high-quality trigger or clock signals from the peripherals to the system timing slot (output).

The DSTAR lines are only available for PXI Express devices when used with a PXI Synchronization Module supporting DSTAR. For more information, refer to the *PXI Express Specification* at `www.pxisa.org`.

9

Bus Interface

The PCI Express/PXI Express interface circuitry of NI 6738/6739 efficiently moves data between host memory and the measurement and acquisition circuits. The digital routing circuitry manages the flow of data between the bus interface and the acquisition/generation sub-systems (analog input, analog output, digital I/O, and the counters), and uses FIFOs (if present) in each sub-system to ensure efficient data movement.

Device Data Transfer Methods

The primary ways to transfer data across the PCI Express/PXI Express bus are as follows:

- **Direct Memory Access (DMA)**—DMA is a method to transfer data between the device and computer memory without the involvement of the CPU. This method makes DMA the fastest available data transfer method. NI uses DMA hardware and software technology to achieve high throughput rates and increase system utilization. DMA is the default method of data transfer for PCI Express and PXI Express devices.

  NI PCI Express and PXI express devices have eight fully-independent DMA controllers for high-performance transfers of data blocks. One DMA controller is available for each measurement and acquisition block:

  – Analog output
  – Counter 0
  – Counter 1
  – Counter 2
  – Counter 3
  – Digital waveform generation (digital output)
  – Digital waveform acquisition (digital input)

  Each DMA controller channel contains a FIFO and independent processes for filling and emptying the FIFO. This allows the buses involved in the transfer to operate independently for maximum performance. Data is transferred simultaneously between the ports. The DMA controller supports burst transfers to and from the FIFO.

  Each DMA controller supports several features to optimize PCI Express/PXI Express bus utilization. The DMA controllers pack and unpack data through the FIFOs. This feature allows the DMA controllers to combine multiple 16-bit transfers to the DAQ circuitry into a single 32-bit burst transfer on PCI Express/PXI Express. The DMA controllers also automatically handle unaligned memory buffers on PCI Express/PXI Express.

- **Programmed I/O**—Programmed I/O is a data transfer mechanism where the user's program is responsible for transferring data. Each read or write call in the program initiates the transfer of data. Programmed I/O is typically used in software-timed (on-demand) operations. Refer to the *Analog Output Data Generation Methods* section of Chapter 4, *Analog Output*, for more information.

PXI Express Considerations

PXI and PXI Express Clock and Trigger Signals

Refer to the *PXIe_DSTAR<A..C>*, *PXIe_CLK100*, and *PXIe_SYNC100* sections of Chapter 8, *Digital Routing and Clock Generation*, for more information about PXI Express clock and trigger signals.

PXI Express

The NI PXIe-6738/6739 can be installed in any PXI Express slot in PXI Express chassis.

PXI Express specifications are developed by the PXI System Alliance (`www.pxisa.org`).

A

Where to Go from Here

This section lists where you can find example programs for the NI 6738/6739 and relevant documentation.

Example Programs

NI-DAQmx software includes example programs to help you get started programming with the NI 6738/6739. Modify example code and save it in an application, or use examples to develop a new application, or add example code to an existing application.

To locate NI software examples, go to `ni.com/info` and enter the Info Code `daqmxexp`. For additional examples, refer to `ni.com/examples`.

To run examples without the device installed, use an NI-DAQmx simulated device. For more information, in Measurement & Automation Explorer (MAX), select **Help»Help Topics» NI-DAQmx»MAX Help for NI-DAQmx** and search for simulated devices.

Related Documentation

Each application software package and driver includes information about writing applications for taking measurements and controlling measurement devices. The following references to documents assume you have NI-DAQmx 15.1 or later.

NI 6738/6739 Documentation

The NI 6738/6739 device specifications are available for download at `ni.com/manuals`.

NI-DAQmx

The *NI-DAQmx Readme* lists which devices, ADEs, and NI application software are supported by this version of NI-DAQmx. Select **Start»All Programs»National Instruments» NI-DAQmx»NI-DAQmx Readme**.

The *NI-DAQmx Help* contains API overviews, general information about measurement concepts, key NI-DAQmx concepts, and common applications that are applicable to all programming environments. Select **Start»All Programs»National Instruments» NI-DAQmx»NI-DAQmx Help**.

LabVIEW

Refer to `ni.com/gettingstarted` for more information about getting started with LabVIEW.

Use the *LabVIEW Help*, available by selecting **Help»LabVIEW Help** in LabVIEW, to access information about LabVIEW programming concepts, step-by-step instructions for using LabVIEW, and reference information about LabVIEW VIs, functions, palettes, menus, and tools. Refer to the following locations on the **Contents** tab of the *LabVIEW Help* for information about NI-DAQmx:

- **VI and Function Reference»Measurement I/O VIs and Functions»DAQmx - Data Acquisition VIs and Functions**—Describes the LabVIEW NI-DAQmx VIs and functions.
- **Property and Method Reference»NI-DAQmx Properties**—Contains the property reference.
- **Taking Measurements**—Contains the conceptual and how-to information you need to acquire and analyze measurement data in LabVIEW, including common measurements, measurement fundamentals, NI-DAQmx key concepts, and device considerations.

LabVIEW NXG

Refer to the *Taking NI-DAQmx Measurements* lessons to assist in getting started in LabVIEW NXG, beginning with *NI-DAQmx API Basics*. To access these lessons, enter `taking NIDAQmx measurements` in the Search bar in LabVIEW NXG.

LabWindows/CVI

The **Data Acquisition** book of the *LabWindows/CVI Help* contains *Taking an NI-DAQmx Measurement in LabWindows/CVI*, which includes step-by-step instructions about creating a measurement task using the DAQ Assistant. In LabWindows™/CVI™, select **Help»Contents**, then select **Using LabWindows/CVI»Data Acquisition**. This book also contains information about accessing detailed information through the *NI-DAQmx Help*.

The **NI-DAQmx Library** book of the *LabWindows/CVI Help* contains API overviews and function reference for NI-DAQmx. Select **Library Reference»NI-DAQmx Library** in the *LabWindows/CVI Help*.

Microsoft Visual Studio Support

You can use the NI-DAQmx .NET class library to communicate with and control an NI data acquisition (DAQ) device. Documentation for the NI-DAQmx .NET class library is available by selecting **Start»All Programs»National Instruments»NI-DAQmx»NI-DAQmx Documentation** and then opening the `NINETDAQmxFxXX.chm` help file corresponding to the version of NI-DAQmx .NET Framework language support you have installed.

- **Measurement Studio Support for NI-DAQmx**—If you program your NI-DAQmxsupported device in Visual Studio using Visual C# or Visual Basic .NET, you can interactively create channels and tasks using Measurement Studio and the DAQ

Assistant. Additionally, you can use Measurement Studio to generate the configuration code based on your task or channel. Refer to the *DAQ Assistant Help* for additional information about generating code.

To create an NI-DAQmx application using Visual Basic .NET or Visual C#, follow these general steps:

1. In Visual Studio, select **File»New»Project** to launch the New Project dialog box.
2. Choose a programming language (Visual C# or Visual Basic .NET), and then select **Measurement Studio** to see a list of project templates.
3. Select **NI DAQ Windows Application**. Choose a project type. You add DAQ tasks as a part of this step.

- **.NET Languages without NI Application Software**—With the Microsoft .NET Framework, you can use the NI-DAQmx .NET class library to create applications using Visual C# and Visual Basic .NET without Measurement Studio. Refer to the *NI-DAQmx Readme* for specific versions supported.

ANSI C without NI Application Software

The *NI-DAQmx Help* contains API overviews and general information about measurement concepts. Select **Start»All Programs»National Instruments»NI-DAQmx»NI-DAQmx Help**.

The *NI-DAQmx C Reference Help* describes the NI-DAQmx Library functions, which you can use with National Instruments data acquisition devices to develop instrumentation, acquisition, and control applications. Select **Start»All Programs»National Instruments»NI-DAQmx» Text-Based Code Support»NI-DAQmx C Reference Help**.

Training Courses

If you need more help getting started developing an application with NI products, NI offers training courses. To enroll in a course or obtain a detailed course outline, refer to `ni.com/training`.

Technical Support on the Web

For additional support, refer to `ni.com/support`.

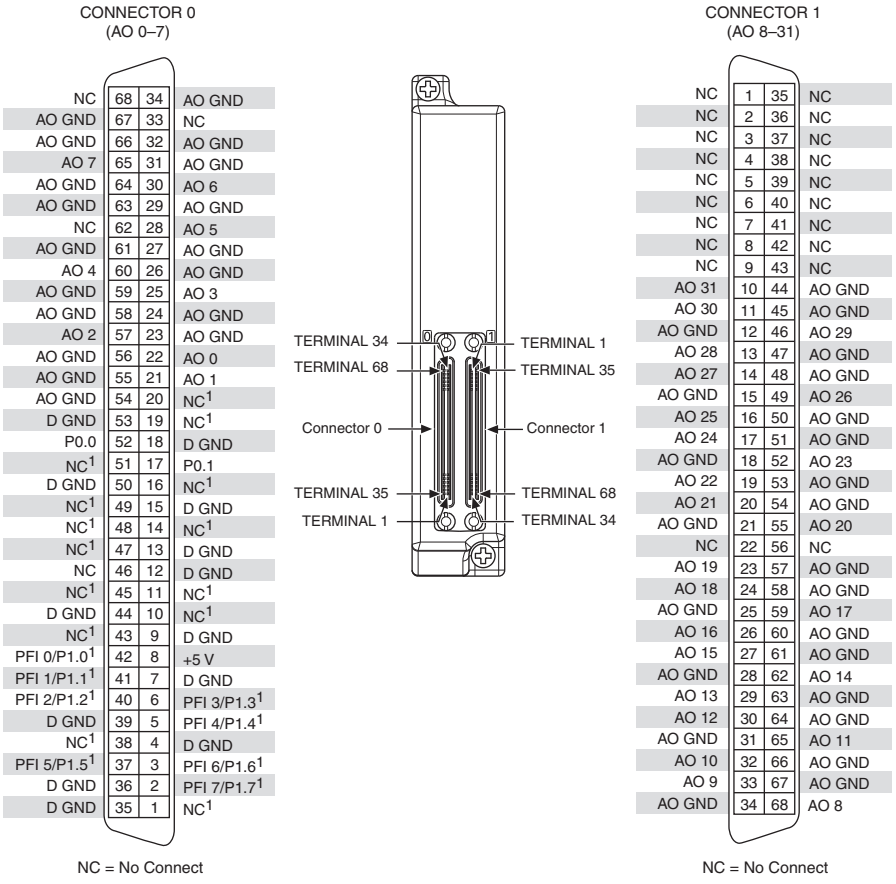
Many DAQ specifications and user guides/manuals are available as PDFs. You must have Adobe Reader 7.0 or later (PDF 1.6 or later) installed to view the PDFs. Refer to the Adobe Systems Incorporated website at `www.adobe.com` to download Adobe Reader. Refer to the National Instruments Product Manuals Library at `ni.com/manuals` for updated documentation resources.

B

NI 6738/6739 in an NI 6723 System (PXI Express Only)

This appendix contains information on connecting your PXIe-6738/6739 to an existing system that is wired and configured for the NI 6723. NI offers adapter accessories you can use, along with SH68-C68-S cables, to incorporate PXIe-6738/6739 signals into a system using the NI 6723 with minimal rewiring and reconfiguration. Figure B-1 shows how to connect to a system configured for the NI 6723 to an PXIe-6738 using the NI 6738 adapter.

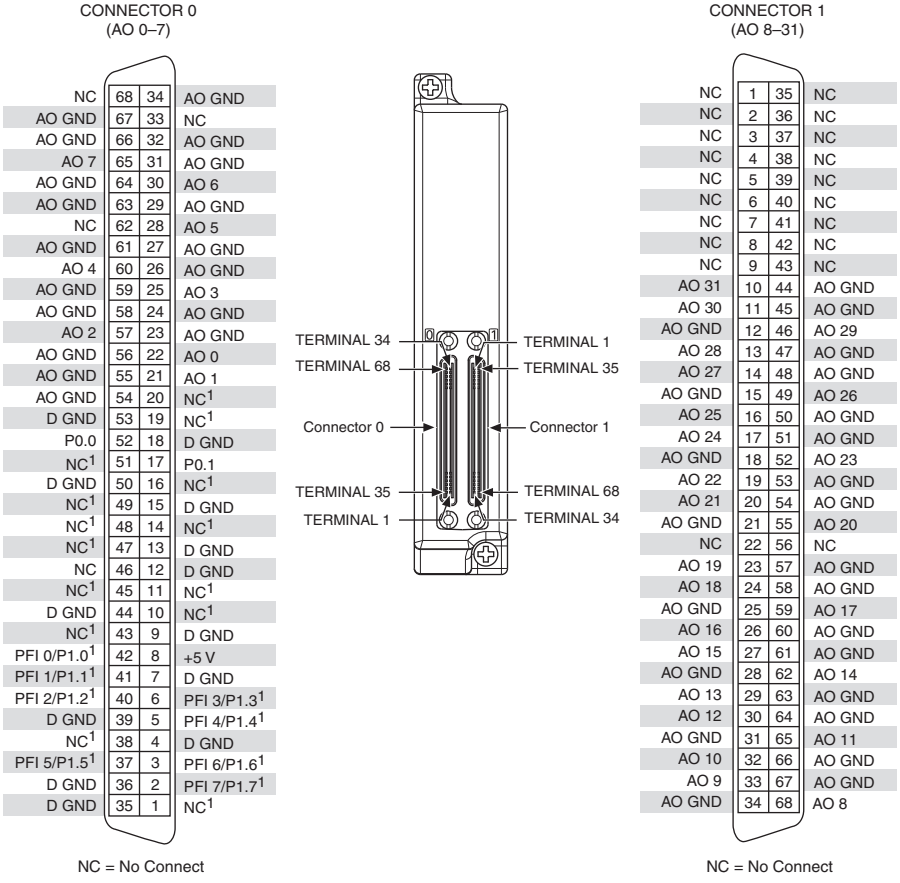
Figure B-1. Connecting to the NI 6723 and to the PXIe-6738 (with Adapter)



1 NI 6723 (with SH68-C68-S Cables) 2 NI 6738 (with Adapter and SH68-C68-S Cables)

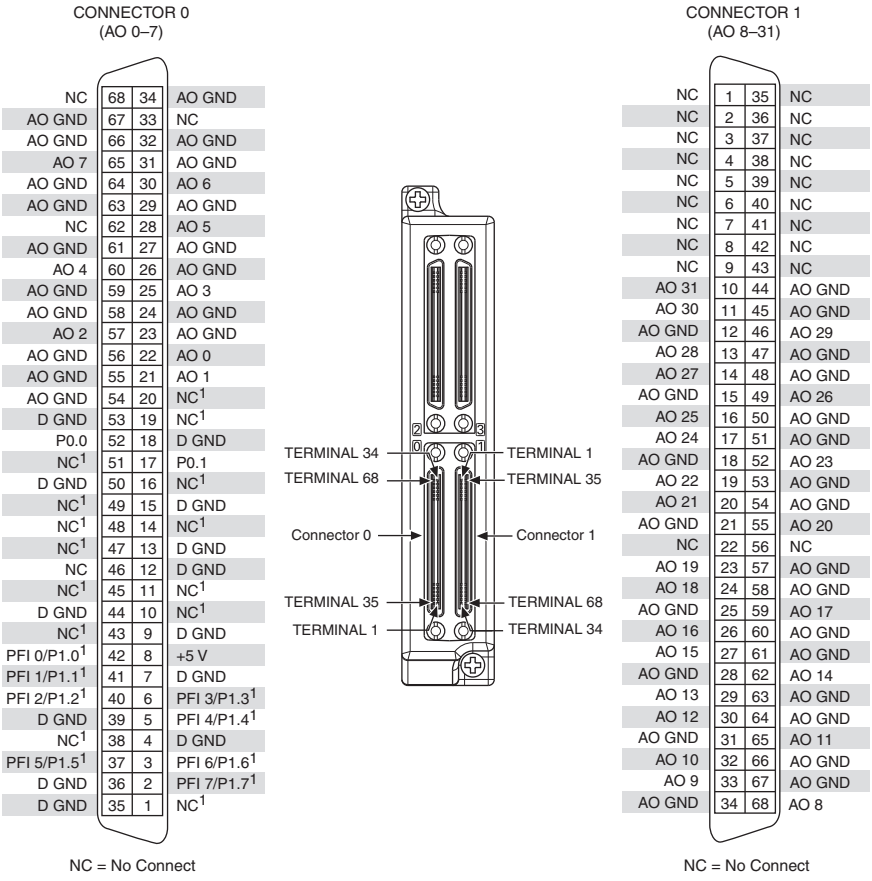
Figure B-2 shows the adapter pinout for using the NI PXIe-6738 in an NI 6723 configuration. Figures B-3 and B-4 show the adapter pinout for using the NI PXIe-6739 in an NI 6723 configuration. The NI 6723 has eight DIO lines, while the NI 6738 and NI 6739 have two and four DIO lines, respectively. The loss of available DIO lines does not affect other DIO characteristics. Refer to your device specifications for digital I/O performance information.

Figure B-2. NI PXIe-6738 Adapter Pinout—Connectors 0 and 1



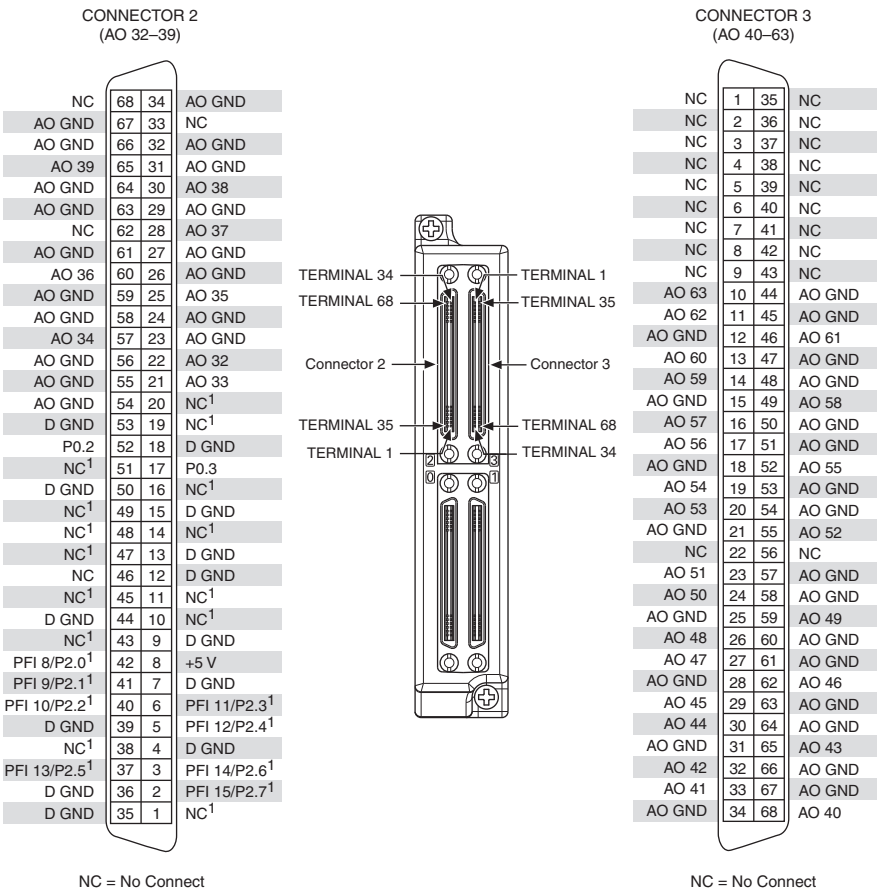
1 Some functionality has been changed or removed as compared to the NI 6723. Refer to the NI 6723 pinout before connecting your system.

Figure B-3. NI PXIe-6739 Adapter Pinout—Connectors 0 and 1



[1] Some functionality has been changed or removed as compared to the NI 6723. Refer to the NI 6723 pinout before connecting your system.

Figure B-4. NI PXIe-6739 Adapter Pinout—Connectors 2 and 3



[1] Some functionality has been changed or removed as compared to the NI 6723. Refer to the NI 6723 pinout before connecting your system.

C

NI Services

National Instruments provides global services and support as part of our commitment to your success. Take advantage of product services in addition to training and certification programs that meet your needs during each phase of the application life cycle; from planning and development through deployment and ongoing maintenance.

To get started, register your product at ni.com/myproducts.

As a registered NI product user, you are entitled to the following benefits:

- Access to applicable product services.
- Easier product management with an online account.
- Receive critical part notifications, software updates, and service expirations.

Log in to your National Instruments ni.com User Profile to get personalized access to your services.

Services and Resources

- **Maintenance and Hardware Services—**NI helps you identify your systems' accuracy and reliability requirements and provides warranty, sparing, and calibration services to help you maintain accuracy and minimize downtime over the life of your system. Visit ni.com/services for more information**.**
  - **Warranty and Repair—**All NI hardware features a one-year standard warranty that is extendable up to five years. NI offers repair services performed in a timely manner by highly trained factory technicians using only original parts at a National Instruments service center.
  - **Calibration—**Through regular calibration, you can quantify and improve the measurement performance of an instrument. NI provides state-of-the-art calibration services. If your product supports calibration, you can obtain the calibration certificate for your product at ni.com/calibration.
- **System Integration**—If you have time constraints, limited in-house technical resources, or other project challenges, National Instruments Alliance Partner members can help. To learn more, call your local NI office or visit ni.com/alliance.

- **Training and Certification**—The NI training and certification program is the most effective way to increase application development proficiency and productivity. Visit ni.com/training for more information.
  - The Skills Guide assists you in identifying the proficiency requirements of your current application and gives you options for obtaining those skills consistent with your time and budget constraints and personal learning preferences. Visit ni.com/skills-guide to see these custom paths.
  - NI offers courses in several languages and formats including instructor-led classes at facilities worldwide, courses on-site at your facility, and online courses to serve your individual needs.
- **Technical Support**—Support at ni.com/support includes the following resources:
  - **Self-Help Technical Resources**—Visit ni.com/support for software drivers and updates, a searchable KnowledgeBase, product manuals, step-by-step troubleshooting wizards, thousands of example programs, tutorials, application notes, instrument drivers, and so on. Registered users also receive access to the NI Discussion Forums at ni.com/forums. NI Applications Engineers make sure every question submitted online receives an answer.
  - **Software Support Service Membership**—The Standard Service Program (SSP) is a renewable one-year subscription included with almost every NI software product, including NI Developer Suite. This program entitles members to direct access to NI Applications Engineers through phone and email for one-to-one technical support, as well as exclusive access to online training modules at ni.com/self-paced-training. NI also offers flexible extended contract options that guarantee your SSP benefits are available without interruption for as long as you need them. Visit ni.com/ssp for more information.
- **Declaration of Conformity (DoC)**—A DoC is our claim of compliance with the Council of the European Communities using the manufacturer's declaration of conformity. This system affords the user protection for electromagnetic compatibility (EMC) and product safety. You can obtain the DoC for your product by visiting ni.com/certification.

For information about other technical support options in your area, visit ni.com/services, or contact your local office at ni.com/contact.

You can also visit the Worldwide Offices section of ni.com/niglobal to access the branch office websites, which provide up-to-date contact information, support phone numbers, email addresses, and current events.