NI Script Editor Help

February 2007, 370777E-01

The National Instruments Script Editor is designed to help you easily create and manipulate scripts. This help file discusses the user interface and features of the Script Editor and explains how to create your own script. This help file also includes information on the instructions and syntax for writing a script.

For more information about this help file, refer to the following topics:

Using Help

Glossary

Important Information

Technical Support and Professional Services

To comment on National Instruments documentation, refer to the National Instruments Web site.

Getting Started

To get started using the Script Editor, choose the item that describes what you want to do from the following list:

- Learn about the scripting language.
- Learn about the Script Editor environment.
- Work with a script.
- Use your script.

To launch the *NI Script Editor Help*, select **Help»NI Script Editor Help** from inside the Script Editor environment.

Using Help

Conventions

This help file uses the following formatting and typographical conventions:

< >	Angle brackets that contain numbers separated by an ellipsis represent a range of values associated with a bit or signal name—for example, AO <0..3>.
[ ]	Square brackets enclose optional items—for example, [response].
»	The » symbol leads you through nested menu items and dialog box options to a final action. The sequence **File»Page Setup»Options** directs you to pull down the **File** menu, select the **Page Setup** item, and select **Options** from the last dialog box.
♦	The ♦ symbol indicates that the following text applies only to a specific product, a specific operating system, or a specific software version.
💡	This icon denotes a tip, which alerts you to advisory information.
📝	This icon denotes a note, which alerts you to important information.
⚠	This icon denotes a caution, which advises you of precautions to take to avoid injury, data loss, or a system crash.
⚡	When symbol is marked on a product, it denotes a warning advising you to take precautions to avoid electrical shock.
♨	When symbol is marked on a product, it denotes a component that may be hot. Touching this component may result in bodily injury.
blue	Text in this color denotes a specific platform and indicates that the text following it applies only to that platform.
bold	Bold text denotes items that you must select or click in the software, such as menu items and dialog box options. Bold text also denotes parameter names.

dark red	Text in this color denotes a caution.
green	Underlined text in this color denotes a link to a help topic, help file, or Web address.
italic	Italic text denotes variables, emphasis, cross-references, or an introduction to a key concept. Italic text also denotes text that is a placeholder for a word or value that you must supply.
monospace	Text in this font denotes text or characters that you should enter from the keyboard, sections of code, programming examples, and syntax examples. This font is also used for the proper names of disk drives, paths, directories, programs, subprograms, subroutines, device names, functions, operations, variables, filenames, and extensions.
monospace bold	Bold text in this font denotes the messages and responses that the computer automatically prints to the screen. This font also emphasizes lines of code that are different from the other examples.
monospace italic	Italic text in this font denotes text that is a placeholder for a word or value that you must supply.

Navigating Help (Windows Only)

To navigate this help file, use the **Contents**, **Index**, and **Search** tabs to the left of this window or use the following toolbar buttons located above the tabs:

- **Hide**—Hides the navigation pane from view.
- **Locate**—Locates the currently displayed topic in the **Contents** tab, allowing you to view related topics.
- **Back**—Displays the previously viewed topic.
- **Forward**—Displays the topic you viewed before clicking the **Back** button.
- **Options**—Displays a list of commands and viewing options for the help file.

Searching Help (Windows Only)

Use the **Search** tab to the left of this window to locate content in this help file. If you want to search for words in a certain order, such as "related documentation," add quotation marks around the search words as shown in the example. Searching for terms on the **Search** tab allows you to quickly locate specific information and information in topics that are not included on the **Contents** tab.

Wildcards

You also can search using asterisk (*) or question mark (?) wildcards. Use the asterisk wildcard to return topics that contain a certain string. For example, a search for "prog*" lists topics that contain the words "program," "programmatically," "progress," and so on.

Use the question mark wildcard as a substitute for a single character in a search term. For example, "?ext" lists topics that contain the words "next," "text," and so on.

Note Wildcard searching will not work on Simplified Chinese, Traditional Chinese, Japanese, and Korean systems.

Nested Expressions

Use nested expressions to combine searches to further refine a search. You can use Boolean expressions and wildcards in a nested expression. For example, "example AND (program OR VI)" lists topics that contain "example program" or "example VI." You cannot nest expressions more than five levels.

Boolean Expressions

Click the ▸ button to add Boolean expressions to a search. The following Boolean operators are available:

- **AND** (default)—Returns topics that contain both search terms. You do not need to specify this operator unless you are using nested expressions.
- **OR**—Returns topics that contain either the first or second term.
- **NOT**—Returns topics that contain the first term without the second term.
- **NEAR**—Returns topics that contain both terms within eight words of each other.

Search Options

Use the following checkboxes on the **Search** tab to customize a search:

- **Search previous results**—Narrows the results from a search that returned too many topics. You must remove the checkmark from this checkbox to search all topics.
- **Match similar words**—Broadens a search to return topics that contain words similar to the search terms. For example, a search for "program" lists topics that include the words "programs," "programming," and so on.
- **Search titles only**—Searches only in the titles of topics.

Printing Help File Topics (Windows Only)

Complete the following steps to print an entire book from the **Contents** tab:

1. Right-click the book.
2. Select **Print** from the shortcut menu to display the **Print Topics** dialog box.
3. Select the **Print the selected heading and all subtopics** option.

   Note Select **Print the selected topic** if you want to print the single topic you have selected in the **Contents** tab.

4. Click the **OK** button.

Printing PDF Documents

This help file may contain links to PDF documents. To print PDF documents, click the print button located on the Adobe Acrobat Viewer toolbar.

Programming Generation Sequences Using Scripting

You can "link and loop" between multiple waveforms in a generation operation using a script. A *script* is a set of instructions that describes the waveforms to be generated, the order in which to generate them, how many times they are generated, and so on. This section describes the syntax for creating a script, as well as how to use scripting in your applications.

A simple script example is shown below.

```
script myFirstScript   generate countUp
  generate allOnes
  generate countDown
end script
```

When executed, this script consecutively generates three waveforms (countUp, allOnes, and countDown).

Scripting Instructions

Scripts consist of six primary instructions: generate, repeat/end repeat, if/else/end if, wait, and clear. Additionally, all instructions in a script are surrounded by the keywords script *<script name>*/end script. Multiple scripts can exist on the device at one time—you can choose which script to execute by referencing the script name.

For examples of scripting applications, refer to Common Scripting Use Cases.

script/end script

Use the script/end script statement to define a set of instructions to be contained within a single script and to associate a name with that script.

Script names must be unique, and they cannot have the same name as a waveform name.

Usages:

- Create a script

  script <*script name*>
    <*instructions*>
  end script

Examples:

- Create a simple script named myScript to generate myWfm:

```
script myScript
  generate myWfm
end script
```

- Create multiple scripts named myScript1 & myScript2:

```
script myScript1
  generate myWfmA
end script

script myScript2
  generate myWfmB
end script
```

Back to Scripting Instructions

generate

Use the generate instruction to describe which waveform to generate.

Usages:

- Generate a waveform written to the device with the Write Named Waveform VI/function for your driver:
  generate <*waveform name*>
- Generate a subset of the named waveform:
  generate <*waveform name*> **subset** (<*start position*>, <*length*>)
  Specify *start position* and *length* in samples.
- Generate the waveform and generate a Marker event when a position(s) within the waveform is generated:
  generate <*waveform name*> **marker0** (*<position 1>*, *<position 2>*, ... , *<position n>*)
  Specify each position in samples. Use the Export Signal VI/function for your driver to specify the destination terminal of the marker. Marker position is zero-based. For example, 0 refers to the first point in the waveform, 999 refers to the 1,000th point in the waveform, and so on.
- Generate a subset of the named waveform and generate a Marker event when a position(s) within the waveform is generated:
  generate *<waveform name>* **subset**(*<start position*), *length>*) **marker0** (*<list of positions>*)
  Specify *positions* and *length* in samples. When a subset and markers are specified in the same generate instruction, the marker positions are relative to the subset.

Examples (assume myWfm has 1,024 samples):

- Generate myWfm:
  generate myWfm
- Generate 10 samples of myWfm starting at sample 40:
  generate myWfm subset (40, 10)
- Generate myWfm and generate a marker at the start of the waveform (sample 0):
  generate myWfm marker0 (0)
- Generate myWfm and generate a marker at positions 10 and 80:
  generate myWfm marker0 (10, 80)

- Generate 10 samples of myWfm starting at sample 40, and generate a marker at position 6 of the subset:

  ```
  generate myWfm subset (40, 10) marker0 (6)
  ```

Back to Scripting Instructions

repeat/end repeat

Use the repeat/end repeat instruction to describe how to "loop" sections of a script until a particular condition is met.

Usages:

- Execute a set of instructions *n* times:
  ```
  repeat <n>
      <instructions>
  end Repeat
  ```
- Execute a set of instructions until the device receives a Script trigger:
  ```
  Repeat until scriptTrigger0
      <instructions>
  end Repeat
  ```
  When the device receives the Script trigger, all instructions remaining in the repeat until loop are completed before advancing, so receipt of the Script trigger does not break out of the repeat loop immediately.
- Execute a set of instructions until the generation operation is aborted (using the Abort VI/function for your driver).
  ```
  repeat forever
      <instructions>
  end repeat
  ```
- Nest repeat <*N*> or repeat until instructions inside a repeat forever instruction:
  ```
  repeat forever
      <instructions>
      repeat <N>
          <instructions>
      end repeat
      <instructions>
  end repeat
  ```

  or:

  ```
  repeat forever
      <instructions>
  ```

```
    repeat until scripttrigger0
        <instructions>
    end Repeat
    <instructions>
end repeat
```

Note You can nest repeat <*N*> and repeat until instructions, one level deep, inside a repeat forever instruction. The if/else/end if instruction is allowed inside a repeat forever instruction, but if/else/end if is not allowed inside of repeat <*N*> and repeat until instructions. Other nesting (for example, repeat <*N*> inside another repeat <*N*>) is not allowed.

Examples:

- Generate myWfmA followed by myWfmB five times:

```
repeat 5
    generate myWfmA
    generate myWfmB
end repeat
```

- Generate the sequence myWfmA, myWfmB, myWfmC until a Script trigger is received:

```
Repeat until scripttrigger0
    generate myWfmA
    generate myWfmB
    generate myWfmC
end repeat
```

- Generate myWfmA forever (until the operation is aborted):

```
repeat forever
    generate myWfmA
end repeat
```

- Generate continuously the sequence initialWfm once; myWfmA, myWfmB, myWfmC 1000 times; myWfmD once:

```
repeat forever
    generate initialWfm
    repeat 1000
        generate myWfmA
        generate myWfmB
        generate myWfmC
    end repeat
    generate myWfmD end repeat
```

- Switch between two waveforms upon receipt of a Script trigger (until the operation is aborted):

```
repeat forever
    repeat until scripttrigger0
        generate myWfmA
    end repeat
    repeat until scripttrigger0
        generate myWfmB
    end repeat
end repeat
```

Back to Scripting Instructions

if/else/end if

Use the if/else/end if instruction to determine what sections of a script to execute, based on whether a particular Script trigger was received.

Note The Script Editor automatically creates an else statement for you, though this statement is optional.

Usage:

- Execute a set of instructions if the device receives a Script trigger, execute another set of instructions otherwise:

  ```
  if scriptTrigger0
      <instructions>
  else
      <instructions>
  end if
  ```

- Execute a set of instructions if the device receives a Script trigger:

  ```
  if scriptTrigger0
      <instructions>
  end if
  ```

- Execute a set of instructions if the device has not received a Script trigger:

  ```
  if scriptTrigger0
  else
      <instructions>
  end if
  ```

Note The if/else/end if instruction must be preceded by a generate or wait <*N*> instruction. You can nest if/else/end if instructions, but if/else/end if instructions are not allowed in repeat until and repeat <*N*> instructions.

Examples:

- Generate myWfmB five times if a Script trigger is received, otherwise generate myWfmC:

  ```
  generate myWfmA
  ```

```
if scripttrigger0
    repeat 5
        generate myWfmB
    end repeat
else
    generate myWfmC
end if
```

- Generate myWfmA if two Script triggers were received:

```
wait 8
if scripttrigger0
    wait 8
    if scripttrigger1
        generate myWfmA
    end if
end if
```

- Generate myWfmA if Script trigger 0 was received, or generate myWfmB if Script trigger 1 was received, otherwise generate myWfmC:

```
wait 8
if scripttrigger0
    generate myWfmA
else
    wait 8

    if scripttrigger1
        generate myWfmB
    else
        generate myWfmC
    end if
end if
```

- If an error is flagged in the form of a Script trigger while generating a waveform, do not generate all subsequent waveforms:

```
generate myWfmA
if scripttrigger0
else
```

```
        generate myWfmB
        if scripttrigger0
        else
            generate myWfmC
        end if
    end if
```

Back to Scripting Instructions

wait

Pause execution of a script. You can pause the script until a particular Script trigger is received or until a specified number of samples are generated.

Usage:

- Pause the execution until a particular Script trigger is received:
  wait until scripttrigger0

  Note If the Script trigger is received before the wait instruction then the script moves to the next instruction with the smallest possible delay. If you want to ignore Script triggers received before a wait statement, use the clear instruction.

- Pause the execution for a finite amount of time:
  wait <number of samples>

Examples:

- Generate myWfmA, wait for receipt of a Script trigger, then generate myWfmB:
```
generate myWfmA
wait until scripttrigger0
generate myWfmB
```
- Generate the sequence myWfmA, myWfmB five times; wait for receipt of a Script trigger; generate myWfmC, myWfmD 10 times; wait for receipt of a Script trigger:
```
repeat forever
    repeat 5
        generate myWfmA
        generate myWfmB
    end repeat
    wait until scripttrigger0
    repeat 10
        generate myWfmC
        generate myWfmD
    end repeat
    wait until scripttrigger0
end repeat
```

- Generate the sequence myWfmA, wait 100 samples, then generate myWfmB:

  ```
  generate myWfmA
  wait 100
  generate myWfmB
  ```

Back to Scripting Instructions

clear

Clear a received Script trigger. The clear instruction is commonly used immediately before a wait or repeat until instruction to ensure that any Script triggers received before the wait or repeat until instructions are ignored.

Usage:

- **clear scripttrigger0**

Examples:

- Generate myWfmA, clear any Script triggers, wait for receipt of a Script trigger, then generate myWfmB:

  ```
  generate myWfmA
  clear scripttrigger0
  wait until scripttrigger0
  generate myWfmB
  ```

  Without the clear instruction, any Script triggers received during generation of myWfmA would cause myWfmB to be generated after the smallest possible delay.
- Continuously step between three waveforms, waiting for a Script trigger between each:

  ```
  repeat forever
      clear scripttrigger0
      wait until scripttrigger0
      generate myWfmA

      clear scripttrigger0
      wait until scripttrigger0
      generate myWfmB

      clear scripttrigger0
      wait until scripttrigger0
      generate myWfmC
  end repeat
  ```

Note Script trigger detectors are cleared automatically when Script triggers are "consumed." For example, you do not need to put a clear instruction between the wait instructions in the following script:

```
generate myWfmA
wait until scripttrigger0
wait until scripttrigger0
generate myWfmB
```

The device waits for two Script triggers to occur before generating myWfmB. The same situation is true when a Script trigger is "consumed" by a repeat until instruction or an if/else/end if instruction.

Back to Scripting Instructions

Common Scripting Use Cases

Single Waveform

```
script upOnly   generate countUp
end script
```

Generating Waveform Subsets

```
script upOnlySubset
  generate countUp subset (10, 40)
end script
```

Note This code snippet generates 40 samples from upOnly, starting at sample 10.

Generating Waveforms with Markers

```
script upOnlyWithMarkers
  generate countUp marker0 (0, 20)
end script
```

Note This code snippet generates the entire countUp waveform and generates a Marker event at samples 0 (the start of the waveform) and 20.

Sequence of Multiple Waveforms

```
script upAllOnesDown
    generate countUp
    generate allOnes
    generate countDown
end script
```

Finite Repetition (*N* Times)

```
script up3AllOnesDown
    generate countUp
    repeat 3
        generate allOnes
    end repeat
    generate countDown
end script
```

Conditional Branching — If/Then Else

```
script upOnesOrDown
    generate countUp
    if scripttrigger0 then
        generate allOnes
    else
        generate countDown
    end else
end script
```

Conditional Repetition — Repeat until Trigger

```
script upAllOnesUntilTrigDown
    generate countUp
    repeat until scripttrigger0
        generate allOnes
    end repeat
generate countDown
end script
```

Continuous Generation — Repeat Forever

```
script upThenUpAndDownForever
    generate countUp
    repeat forever
        generate countUpAndDown
    end repeat
end script
```

Waiting for Triggers

```
script upWaitAllZerosDown
    generate countUp
    wait until scripttrigger0
    generate allZeros
    generate countDown
end script
```

or

```
script upWaitAllZerosDown
    generate countUp
    clear scripttrigger0
    wait until scripttrigger0
    generate allZeros
    generate countDown
end script
```

Note These two scripts are similar, but a script received during generation of countUp causes the first script to move to allZeros after the smallest possible delay. By adding a clear instruction, you can ignore any triggers received before the wait instruction.

Finite Wait

```
script upWait32Down
    generate countUp
    wait 32
    generate countDown
end script
```

Stepping Through Multiple Waveforms

```
script stepThroughUpAllZerosDown
    repeat forever
        generate countUp
        clear scripttrigger0
        wait until scripttrigger0

        generate allZeros
        clear scripttrigger0
        wait until scripttrigger0

        generate countDown
        clear scripttrigger0
        wait until scripttrigger0

    end repeat
end script
```

Bursting through Multiple Waveforms

```
script burstThroughUpDownThenZerosOnes
    repeat forever
        repeat until scripttrigger0
            generate countUp
            generate countDown
        end repeat

        repeat until scripttrigger0
            generate allZeros
            generate allOnes
        end repeat
    end repeat
end script
```

Learning about the Script Editor Environment

The following topics provide information about features, terminology, and controls of the Script Editor environment.

Click one of the items below for more information about that topic. You can also use the links within the topics to navigate through the thread.

- Toolbars
- Selecting a Script
- Script Syntax Status
- Instruction Settings

Back to Getting Started

Toolbar

The Script Editor Toolbar includes buttons for the most common commands you use when creating a script:

New—Creates a new script file. If an existing script file is open when you select **New**, you are prompted to either save or close without saving the existing file.

Open—Opens an existing script file.

Save—Saves the script file that is currently open. If a script file is already saved under the specified name, this command overwrites it with the current script file.

Add Generate—Adds a generate instruction to the current script.

Add Repeat—Adds a repeat instruction to the current script.

Add If/Else/End If—Adds an if else end if instruction to the current script.

Add Wait—Adds a wait instruction to the current script.

Add Clear—Adds a clear instruction to the current script.

Delete Instruction—Deletes highlighted instruction(s).

Next topic: Selecting a Script

Selecting a Script

Each script file may contain multiple scripts. You can view a particular script by highlighting the script name in the Script window at the top-right corner of the Script Editor environment.

Next topic: Script Syntax Status

Script Syntax Status

The script syntax status is displayed in the Syntax window at the lower-right corner of the Script Editor environment. The The statusports if the highlighted script contains syntax errors. The first instruction in the script file that contains an error has an **X** to the left of it.

The indicator at the bottom of the window denotes the status of the currently displayed script.

Next topic: Instruction Settings

Instruction Settings

Selecting **Instructions»Add Generate**, **Instructions»Add Repeat**, or clicking the **Generate** or **Repeat** buttons on the toolbar launches a dialog box where you can configure settings for that instruction. The setting options vary by instruction. When an instruction is highlighted within the script, these settings are displayed in the middle section of the right side of the Script Editor window.

Back to Learning About the Script Editor Environment

Working with Scripts

Creating a New Script

Script files may contain multiple scripts. The Script Editor opens with an untitled script file with one script, `myScript1`. To add another script to the file, go to **Instructions»Add Script** and specify a name for the new script.

Opening an Existing Script File

To open an existing script, select **File»Open**, or click the **Open** button on the toolbar, and browse to the script you want to open. If you have an unsaved script that is open, the Script Editor prompts you to save the file.

Renaming a Script

The names of all the scripts within the script file are displayed at the upper-right corner of the Script window. To rename a particular script, highlight the name and right-click to select **Rename Script**.

Next topic: Adding Instructions

Back to Getting Started

Adding Instructions

These options are all available by clicking the add instruction buttons on the toolbar.

Generate

Generates a named waveform. You can generate the entire waveform or a subset of it. You can also choose to generate a marker at a specified location in the named waveform. For example, to generate a marker at the 150th sample of the waveform, type 150 in the **Marker 0** textbox.

Repeat

Repeats a set of instructions. You can repeat a finite number of times, repeat forever, or repeat until a hardware Script trigger is received.

If/Else/End If

Determines what sections of a script to execute, based on whether a particular Script trigger was received.

Wait

Pauses execution until a Script trigger is received.

Clear

Ignores a previously latched Script trigger. This option is typically used before the wait or repeat until instructions. For example, if the two instructions in myScript are:

```
script myScript
  generate mywfm
  wait until scriptTrigger0
end script
```

The Script trigger may arrive when mywfm is being generated. So when Wait until scriptTrigger0 is called, it is executed after the shortest possible delay. If you wish to ignore previously asserted Script triggers, then use the clear instruction:

```
script myScript
  generate mywfm
  clear scriptTrigger0
  wait until scriptTrigger0
end script
```

Next topic: Editing Scripts

Editing Scripts

Modifying Instruction Properties

When an instruction is highlighted within the script, the settings for that instruction are displayed in the Instruction Settings section of the Script Editor environment. You can change the instruction settings by entering new values in the controls in the Instruction Settings section.

Moving Instructions

You can move an instruction by highlighting it and holding down the left mouse button and dragging the instruction to the desired location.

Next topic: Saving Scripts

Saving Scripts

You can save your script either as a text file or as a nonexecutable binary file.

Text Files

To save your script file as a new script file, select **File»Save** or click the **Save** button on the toolbar and specify the file name.

To save your script file with a new name, select **File»Save As** to open the Save As dialog box. Enter a name for the new file and click **OK** to save.

You can use this script file later in your application development environment (ADE) to manage waveform generation.

Nonexecutable Files

You can save non-executable binary files, or script files with errors, so you can later edit the script.

Back to Working with Scripts

Using Your Script

You can use your script in an NI ADE, such as LabVIEW or LabWindows/CVI, in conjunction with your NI instrument driver for the functions/VIs to manage waveform generation.

To use your script, you can either save it and then open it in your ADE; or you can select **Edit»Export to Clipboard**, open your ADE, and then paste the clipboard contents into the appropriate VI.

Back to Getting Started

Glossary

Numbers/Symbols A B C D E F H I L M P R S T V W

Numbers and Symbols

-	negative of, or minus
<	less than
>	greater than
≤	less than or equal to
≥	greater than or equal to
/	per
%	percent
±	plus or minus

A

ADE	Application Development Environment
API	Application Programming Interface—a standardized set of subroutines or functions, along with the parameters that a program can call.

B

b	bits
B	bytes
buffer	1. Temporary storage for acquired or generated data (software). 2. A collection of samples.

C

clock	1. Hardware component that controls timing for reading from or writing to channels. 2. Periodic digital edges that can be used to measure time.
counter/timer	A circuit that counts external pulses or clock pulses (timing).

D

DAQ	Data Acquisition—Collecting and measuring electrical signals from sensors, transducers, and test probes or fixtures and inputting them to a computer for processing. Also refers to collecting and measuring the same kinds of electrical signals with analog-to-digital and/or digital devices plugged into a PC, and possibly generating control signals with digital-to-analog and/or digital devices in the same PC.
default setting	Default parameter value recorded in the driver. In many cases, the default input of a control is a certain value (often 0) that means use the current default setting.
device	Plug-in data acquisition board, card, or pad that can contain multiple channels and conversion devices. Plug-in boards, PCMCIA cards, and devices that connects to your computer parallel port, are all examples of DAQ devices.
digital trigger	Level signal having two discrete levels: a high and a low level. See trigger.
DIO	digital input/output

E

event	Events are emitted to signify a device state change, the arrival of a certain kind of sample, the production of a certain number of samples, or the passage of time.

F

function	Set of software instructions executed by a single line of code that can have input and/or output parameters and returns a value when executed.

H

high level	For generation, the high level is the voltage produced when a binary one is generated. For acquisition, the high level is the voltage threshold above which the input will be sampled as a binary one.

I

I/O	input/output—Transfer of data to/from a computer system involving communications channels, operator interface devices, and/or data acquisition and control interfaces.
idle state	Specifies the values of the channels when the generation operation is paused or has completed.
initial state	Specifies the values of the channels when the generation operation has not yet started.
instructions	Statements used to define a script.

L

latch	
line	Represents the value of one bit of a sample over all samples. A line is independent of any hardware I/O connector.
low level	For generation, the low level is the voltage produced when a binary zero is generated. For acquisition, the low level is the voltage threshold below which the input will be sampled as a binary one.
LSB	least significant bit

M

Marker event	An event that the device generates in relation to a waveform that is generated. You can configure the position(s) at which Marker events are generated.
MB/s	Unit for data transfer that means one million or 2^{20} bytes per second.

P

Pause trigger	Trigger used to indicate to the device that it should stop generating and/or acquiring. The device resumes when the pause trigger becomes inactive.
posttrigger	Acquiring data that occurs after a trigger.
pretrigger	Acquiring data that occurs before a trigger.
propagation delay	The amount of time required for a signal to pass through a circuit.

R

reference clock	Clock to which a device phase locks another, usually faster, clock. A common source for the reference clock is the 10 MHz oscillator present on the PXI backplane.

S

s	seconds
S	sample
sample	The value being generated/acquired on all channels during a single sample clock cycle.
Sample clock	Samples are generated or acquired based on Sample clock cycles.
script	Collection of instructions that describe the order and timing of one or more waveforms.
Script trigger	General-purpose trigger that has a role that is determined by the context of the script.
software trigger	Programmed event that triggers an operation such as data acquisition.

T

terminal	Named location where a signal is either produced (generated) or consumed (acquired).
transfer rate	Rate, measured in bytes/s or samples/s, at which data is moved from source to destination after software initialization and set up operations; the maximum rate at which the hardware can operate.
trigger	A signal sent to the device to control the device in some way. In the context of the NI digital waveform generator/analyzer, triggers are essentially the opposite of events.

V

VI	Virtual Instrument 1. A combination of hardware and/or software elements, typically used with a PC, that has the functionality of a classic stand-alone instrument. 2. A LabVIEW software module (VI), which consists of a front panel user interface and a block diagram program.
virtual channels	Channel names that can be defined outside the application and used without having to perform scaling operations.

W

waveform A collection of samples generated or acquired on a channel.

Important Information

Warranty

Copyright

Trademarks

Patents

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

Technical Support and Professional Services

Visit the following sections of the National Instruments Web site at ni.com for technical support and professional services:

- Support—Online technical support resources at ni.com/support include the following:
  - **Self-Help Resources**—For answers and solutions, visit the award-winning National Instruments Web site 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.
  - **Free Technical Support**—All registered users receive free Basic Service, which includes access to hundreds of Applications Engineers worldwide in the NI Discussion Forums at ni.com/forums. National Instruments Applications Engineers make sure every question receives an answer.

    For information about other technical support options in your area, visit ni.com/services or contact your local office at ni.com/contact.
- Training and Certification—Visit ni.com/training for self-paced training, eLearning virtual classrooms, interactive CDs, and Certification program information. You also can register for instructor-led, hands-on courses at locations around the world.
- 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.

If you searched ni.com and could not find the answers you need, contact your local office or NI corporate headquarters. You also can visit the Worldwide Offices section of ni.com/niglobal to access the branch office Web sites, which provide up-to-date contact information, support phone numbers, email addresses, and current events.

Branch Offices

Office	Telephone Number
Australia	1800 300 800
Austria	43 662 457990-0
Belgium	32 (0) 2 757 0020
Brazil	55 11 3262 3599
Canada	800 433 3488
China	86 21 6555 7838
Czech Republic	420 224 235 774
Denmark	45 45 76 26 00
Finland	385 (0) 9 725 72511
France	33 (0) 1 48 14 24 24
Germany	49 89 7413130
India	91 80 41190000
Israel	972 0 3 6393737
Italy	39 02 413091
Japan	81 3 5472 2970
Korea	82 02 3451 3400
Lebanon	961 (0) 1 33 28 28
Malaysia	1800 887710
Mexico	01 800 010 0793
Netherlands	31 (0) 348 433 466
New Zealand	0800 553 322
Norway	47 (0) 66 90 76 60
Poland	48 22 3390150
Portugal	351 210 311 210
Russia	7 495 783 6851
Singapore	1800 226 5886
Slovenia	386 3 425 42 00

South Africa	27 0 11 805 8197
Spain	34 91 640 0085
Sweden	46 (0) 8 587 895 00
Switzerland	41 56 2005151
Taiwan	886 02 2377 2222
Thailand	662 278 6777
Turkey	90 212 279 3031
United Kingdom	44 (0) 1635 523545
United States (Corporate)	512 683 0100