Model 675

High Performance Arbitrary Waveform Generator

TrueArb Mode Programming Manual

(Active Technology AWG-4010 Series)

Rev 1.4, Dec 2020

TABLE OF CONTENTS

1. PREFACE 8
1.1 ABBREVIATIONS AND TERMS 8
1.2 REVISION HISTORY 8
2. SYNTAX AND COMMANDS 12
2.1 COMMAND SYNTAX 12
2.1.1 Syntax Overview 12
2.1.2 Command and Query Structure 12
2.1.3 Command Entry 14
2.1.4 Parameter Types 15
2.1.5 SCPI Commands and Queries 18
2.2 STATUS AND EVENTS 19
2.2.1 Status and event reporting system 19
2.2.2 Status Byte Register (SBR) 21
2.2.3 Service Request Enable Register (SRER) 22
2.2.4 Standard Event Status Block (SESB) 22
2.2.5 Operation status block 23
2.2.6 Questionable status block 24
2.3 ANALOG DATA FORMAT (.TXT FILE ONLY) 25
2.4 DIGITAL DATA FORMAT (.TXT FILE ONLY) 25
2.5 GRANULARITY 26
2.6 TRANSFERRING DATA FILE 26
2.6.1 Block Data Format 26
2.7 BYTE ORDER DURING TRANSFER 26
2.8 HOW TO GENERATE AN ARBITRARY WAVEFORM 26
2.9 COMMAND GROUPS 28
2.9.1 Control group commands 28
2.9.2 Calibration and Diagnostic 29
2.9.3 Output Group Commands 29
2.9.4 Display Commands 30
2.9.5 License Commands 30
2.9.6 Marker Commands 31
2.9.7 Clock Group Commands 31
2.9.8 IEEE Mandated and Optional Group Command 31
2.9.9 Status Group Command 32
2.9.10 System Group Commands 33
2.9.11 Memory Group Commands 33
2.9.12 MASS Memory Commands 34

2.9.13 Trigger Group Commands 35
2.9.14 Sequence Group Commands 36
2.9.15 Waveform Group Commands 37
2.9.16 Multi Instrument Commands 38
2.10 CONTROL GROUP COMMANDS 39
2.11 CALIBRATION AND DIAGNOSTIC COMMANDS 47
2.12 OUTPUT GROUP COMMANDS 48
2.13 DISPLAY GROUP COMMANDS 53
2.14 LICENSE GROUP COMMANDS 56
2.15 MARKER GROUP COMMANDS 58
2.16 CLOCK GROUP COMMANDS 60
2.17 IEEE MANDATED AND OPTIONAL GROUP COMMANDS 61
2.18 MEMORY GROUP COMMANDS 65
2.19 MASS MEMORY GROUP COMMANDS 70
2.20 STATUS GROUP COMMANDS 86
2.21 SYSTEM GROUP COMMANDS 90
2.22 TRIGGER GROUP COMMANDS 93
2.23 SEQUENCE GROUP COMMANDS 96
2.24 WAVEFORM GROUP COMMANDS 108
2.25 MULTI INSTRUMENT GROUP COMMANDS 115
3. COMMAND ERRORS 117
4. PREDEFINED WAVEFORMS 120
5. REMOTE CONTROL 121
5.1 PREREQUISITE 121
5.1.1 AT Instrument Communicator 126
5.2 NI LABVIEW EXAMPLES 128
5.2.1 Continuous Mode 129
5.2.2 Burst Mode 130
5.2.3 Stepped Mode 132
5.2.4 Advanced Mode 134
5.2.5 Import an Arbitrary Waveform Generation 136
5.3 SCRIPT EXAMPLES 140
5.3.1 Continuous Mode 140
5.3.2 Stepped Mode 141
5.3.3 Import Arbitrary 142
5.3.4 Advanced Mode 143

LIST OF TABLES

Table 1: Abbreviations and terms 8
Table 2: Revision History 11
Table 3: Syntax symbols and their meanings 12
Table 4: Message symbols and their meanings 13
Table 5: Message terminator and meaning 15
Table 6: Parameter types, their descriptions, and examples 16
Table 7: String symbol and meaning 16
Table 8: SI prefixes and their indexes 17
Table 9: Status Byte Register (SBR) 21
Table 10: Service Request Enable Register (SRER) 22
Table 11: Standard Event Status Register (SESR) 23
Table 12: Operation Condition Register (OCR) 24
Table 13: Models and available parameters 28
Table 14: Control group commands 29
Table 15: Calibration and Diagnostic group commands 29
Table 16: Output group commands 30
Table 17: Display group commands 30
Table 18: License group commands 30
Table 19: Marker group commands 31
Table 20: Clock group commands 31
Table 21: IEEE Mandatory group commands 32
Table 22:Status group commands 33
Table 23: System group commands 33
Table 24: Memory Group commands 34
Table 25: Mass Memory Group commands 35
Table 26: Trigger group commands 35
Table 27: Sequence group commands 37
Table 28: Waveform group commands 38
Table 29: Multi-Instrument group commands and their descriptions 38
Table 30: AWGControl:AFGSwitch 39
Table 31: AWGControl:CONFigure:CNUMber 40
Table 32: AWGControl:CONFigure:DNUMber 40
Table 33: AWGControl:DECreasing 41
Table 34: AWGControl:INCreasing 41
Table 35: AWGControl:LENGth:MODE 42
Table 36: AWGControl:RESET[:IMMediate] 42
Table 37: AWGControl:RMODe 43

Table 38: AWGControl:RSTATe 44
Table 39: AWGControl:RUN[:IMMediate] 44
Table 40: AWGControl:SREStore 45
Table 41: AWGControl:SSAVe 45
Table 42: AWGControl:STOP[:IMMediate] 45
Table 43: AWGControl:WAITstate 46
Table 44: AWGControl:JUMPMode 46
Table 45: AWGControl:DJStrobe 47
Table 46: CALibration[:ALL] 47
Table 47: DIAGnostic[:ALL] 48
Table 48: OUTPut[n]:BLOFfset 48
Table 49: OUTPut[n]:DELay 49
Table 50: OUTPut[n]:POLarity 49
Table 51: OUTPut[n]:SCALe 50
Table 52: OUTPut[n]:SERIESIMPedance 50
Table 53: OUTPut[n][:STATe] 51
Table 54: DIGitals:LEVel[m] 51
Table 55: DIGitals:NUMber 52
Table 56: DIGitals:SKEW[m] 53
Table 57: DIGitals:STATe 53
Table 58: DISPlay:FOCus 54
Table 59: DISPlay:UNIT:VOLT 54
Table 60: DISPlay[:WINDow]:TEXT:CLEar 55
Table 61: DISPlay[:WINDow]:TEXT[:DATA] 55
Table 62: HCOPy:SDUMp[:IMMediate] 56
Table 63: LICense:ERRor 56
Table 64: LICense:HID 56
Table 65: LICense: INSTall 57
Table 66: LICense:LIST 57
Table 67: *OPT 58
Table 68: MARKer:LEVel[m] 58
Table 69: MARKer:MODE[m] 59
Table 70: MARKer:SKEW[m] 60
Table 71: ROSCillator 60
Table 72: AWGControl:SRATe 61
Table 73: ROSCillator:SOURce 61
Table 74: *CAL 62
Table 75: *CLS 62
Table 76: *ESE 62

Table 77: *ESR 63
Table 78: *IDN 63
Table 79: *OPC 64
Table 80: *RST 64
Table 81: *SRE 64
Table 82: *TRG 65
Table 83: *TST 65
Table 84: *WAI 65
Table 85: *RCL 66
Table 86: *SAV 66
Table 87: MEMory:NSTates 67
Table 88: MEMory:STATe:CATalog 67
Table 89: MEMory:STATe:DELete 68
Table 90: MEMory:STATe:LOCK 68
Table 91: MEMory:STATe:NAME 69
Table 92: MEMory:STATe:VALid 69
Table 93: DELete:SETUp 69
Table 94: RECALL:SETUp 70
Table 95: MMEMory:CDIRectory 71
Table 96: MMEMory:COPY 72
Table 97: MMEMory:DATA 73
Table 98: MMEMory:DATA:SIZE 74
Table 99: MMEMory:DELete 74
Table 100: MMEMory:DOWNload:DATA 75
Table 101: MMEMory:DOWNload:FNAMe 75
Table 102: MMEMory:EXPort 76
Table 103: MMEMory:IMPort 77
Table 104: MMEMory:LOAD:ALL 78
Table 105: MMEMory:LOAD:STATe 79
Table 106: MMEMory:LOAD:STATe 80
Table 107: MMEMory:LOAD:STATe 81
Table 108:MMEMory:MSIS 81
Table 109: MMEMory:OPEN 82
Table 110: MMEMory:OPEN:SETup 83
Table 111: MMEMory:RDIRectory 83
Table 112: MMEMory:STORe:ALL 84
Table 113: MMEMory:SAVE:SETup 84
Table 114: MMEMory:STORe:STATe 85
Table 115: MMEMory:UPLoad 86

Table 116: STATus:OPERation:CONDition 86
Table 117: STATus:OPERation:ENABle 87
Table 118: STATus:OPERation[:EVENt] 87
Table 119: STATus:PRESet 87
Table 120: STATus:QUEStionable:CONDition 88
Table 121: STATus:QUEStionable:ENABle 88
Table 122: STATus:QUEStionable[:EVENt] 88
Table 123: *STB 89
Table 124: *PSC 89
Table 125: SYSTem:BEEPer:STATe 90
Table 126: SYSTem:BEEPer[:IMMediate] 90
Table 127: SYSTem:DATE 91
Table 128: SYSTem:ERRor[:NEXT] 91
Table 129: SYSTem:KLOCk[:STATe] 92
Table 130: SYSTem:SECurity:IMMediate 92
Table 131: SYSTem:TIME 92
Table 132: SYSTem:VERSion 93
Table 133: ABORt 93
Table 134: TRIGger[:SEQuence]:SOURce 94
Table 135: TRIGger[:SEQuence]:SLOPe 94
Table 136: TRIGger[:SEQuence]:LEVel 95
Table 137: TRIGger[:SEQuence]:TIMer 95
Table 138: TRIGger:IMPedance 96
Table 139: TRIGger[:SEQuence][:IMMediate] 96
Table 140: SEQuence:ELEM[n]:AMPlitude[m] 97
Table 141: SEQuence:ELEM[n]:OFFset[m] 97
Table 142: SEQuence:ELEM[n]:VOLTage:HIGH[m] 98
Table 143: SEQuence:ELEM[n]:VOLTage:LOW[m] 98
Table 144: SEQuence:LENgth 99
Table 145: SEQuence:ELEM[n]:LOOP:COUNt 100
Table 146: SEQuence:ELEM[n]:WAVeform[m] 100
Table 147: SEQuence:LENGth 101
Table 148: SEQuence:NEW 101
Table 149: SEQuence:FOCus 101
Table 150: SEQuence:ELEM[n]:WAITEvent 102
Table 151: SEQuence:ELEM[n]:GOTOMode 103
Table 152: SEQuence:ELEM[n]:GOTOEntry 103
Table 153: SEQuence:ELEM[n]:JUMPTOMode 104
Table 154: SEQuence:ELEM[n]:JUMPEvent 105

Table 155: SEQuence:ELEM[n]:JUMPTOEntry 106
Table 156: SEQuence:ELEM[n]:PATTERN 106
Table 157: SEQuence:ELEM[n]:PATTERNJUMPTOMode 107
Table 158: SEQuence:ELEM[n]:PATTERNJUMPTOEntry 108
Table 159: WLISt:LIST 109
Table 160: WLISt:NAME 109
Table 161: WLISt:SIZE 109
Table 162: WLISt:WAVeform:DATA 111
Table 163: WLISt:WAVeform:DELete 111
Table 164: WLISt:WAVeform:IMPort 113
Table 165: WLISt:WAVeform:LMAXimum 113
Table 166: WLISt:WAVeform:LMINimum 113
Table 167: WLISt:WAVeform:LENGth 114
Table 168: WLISt:WAVeform:PREDefined 114
Table 169: WLISt:WAVeform:TYPE 114
Table 170: MIM:CAPTure 115
Table 171: MIM:ID 115
Table 172: MIM:CAPTured 116
Table 173: MIM:FORWard 116
Table 174: MIM:SLAve 116
Table 175: MIM:NUMber 117
Table 176: MIM:RELease 117
Table 177: Command Errors 120
Table 178: Predefined Waveforms 120

1. PREFACE

Scope of this document is to describe the use of SCPI commands with the Model 675 series when used in the True-Arb Operating Mode.

1.1 Abbreviations and terms

Abbreviation	Description
SW	Software
UI	User Interface
API	Application Programming Interface
FG	Function Generator
AM	Amplitude Modulation
FM	Frequency Modulation
PM	Phase Modulation
PWM	Pulse Width Modulation
SCPI	Standard Commands for Programmable Instruments
AWG	Arbitrary Waveform Generator
SDK	Software Development Kit
VISA	Virtual Instrument Software Architecture

Table 1: Abbreviations and terms

1.2 Revision History

Rev.	Document Changes	Date

1.3	**Modified Commands**: AWGControl:RMODe **Added commands**: AWGControl:JUMPMode {AFTERrepetitions | IMMediate} AWGControl:JUMPMode {AFTERrepetitions | IMMediate} AWGControl:JUMPMode? SEQuence:ELEM[#]:WAITEvent {NONE | MANual | TIMer | EXTernal} SEQuence:ELEM[#]:WAITEvent? SEQuence:ELEM[#]:GOTOMode {FIRST | PREVious | NEXT | LAST | ITEM} SEQuence:ELEM[#]:GOTOMode? SEQuence:ELEM[#]:GOTOEntry {MINimum | MAXimum | DEFault | <value>} SEQuence:ELEM[#]:GOTOEntry? [{MINimum | MAXimum}] SEQuence:ELEM[#]:JUMPTOMode {FIRST | PREVious | NEXT | LAST | ITEM} SEQuence:ELEM[#]:JUMPTOMode? SEQuence:ELEM[#]:JUMPEvent {NONE | MANual | TIMer | EXTernal} SEQuence:ELEM[#]:JUMPEvent? SEQuence:ELEM[#]:JUMPTOEntry {MINimum | MAXimum | DEFault | <value>} SEQuence:ELEM[#]:JUMPTOEntry? [{MINimum | MAXimum}] SEQuence:ELEM[#]:PATTERN {MINimum | MAXimum | DEFault | <value>} SEQuence:ELEM[#]:PATTERN? [{MINimum | MAXimum}] SEQuence:ELEM[#]:PATTERNJUMPTOMode {FIRST | PREVious | NEXT | LAST | ITEM} SEQuence:ELEM[#]:PATTERNJUMPTOMode? SEQuence:ELEM[#]:PATTERNJUMPTOEntry {MINimum | MAXimum | DEFault | <value>}	May 7, 2019

	SEQuence:ELEM[#]:PATTERNJUMPTOEntry? [{MINimum | MAXimum}] AWGControl:DJStrobe MIM:CAPture MIM:RELease MIM:ID? MIM:SLAve? MIM:FORWard? MIM:CAPTured? MIM:NUMber?	
1.4	**Errata Corrige Manual** OUTPut[n][:STATe] DIGitals:NUMber add "s" of "DIGitals"	April 17, 2020

Table 2: Revision History

2. SYNTAX AND COMMANDS

2.1 Command Syntax

2.1.1 Syntax Overview

Control the operations and functions of the instrument through the LAN interface using commands and queries. The related topics listed below describe the syntax of these commands and queries. The topics also describe the conventions that the instrument uses to process them. See the Command Groups topic for a listing of the commands by command group or use the index to locate a specific command.
Refer to the following table for the symbols that are used.

Symbol	Meaning
< >	Defined element
::=	Is defined as
|	Exclusive OR
{ }	Group; one element is required
[ ]	Optional; can be omitted
...	Previous elements can be repeated
( )	Comment

Table 3: Syntax symbols and their meanings

2.1.2 Command and Query Structure

Overview

Commands consist of set commands and query commands (usually called commands and queries). Commands modify instrument settings or tell the instrument to perform a specific action. Queries cause the instrument to return data and status information.
Most commands have both a set form and a query form. The query form of the command differs from the set form by its question mark on the end.
For example, the set command *OUTPut1:STATe* has a query form *OUTPut1:STATe?*.
Not all commands have both a set and a query form. Some commands have only set and some have only query.

Messages

A command message is a command or query name followed by any information the instrument needs to execute the command or query. Command messages may contain five element types, defined in the following table.

Symbol	Meaning
<Header>	This is the basic command name. If the header ends with a question mark, the command is a query. The header may begin with a colon (:) character. If the command is concatenated with other commands, the beginning colon is required. Never use the beginning colon with command headers beginning with a star (*).
<Mnemonic>	This is a header subfunction. Some command headers have only one mnemonic. If a command header has multiple mnemonics, a colon (:) character always separates them from each other.
<Argument>	This is a quantity, quality, restriction, or limit associated with the header. Some commands have no arguments while others have multiple arguments. A <space> separates arguments from the header. A <comma> separates arguments from each other.
<Comma>	A single comma is used between arguments of multiple-argument commands. Optionally, there may be white space characters before and after the comma.
<Space>	A white space character is used between a command header and the related argument. Optionally, a white space may consist of multiple white space characters.

Table 4: Message symbols and their meanings

Commands

Commands cause the instrument to perform a specific function or change one of the settings. Commands have the structure:
[:]<Header>[<Space><Argument>[<Comma><Argument>]...]
A command header consists of one or more mnemonics arranged in a hierarchical or tree structure. The first mnemonic is the base or root of the tree and each subsequent mnemonic is a level or branch of the previous one. Commands at a higher level in the tree may affect those at a lower level. The leading colon (:) always returns you to the base of the command tree.

Queries

Queries cause the instrument to return status or setting information.
Queries have the structure:
[:]<Header>?
[:]<Header>?[<Space><Argument>[<Comma><Argument>]...]

2.1.3 Command Entry

Rules

The following rules apply when entering commands:

- You can enter commands in upper or lower case.
- You can precede any command with white space characters. White space characters include any combination of the ASCII control characters 00 through 09 and 0B through 20 hexadecimal (0 through 9 and 11 through 32 decimal).
- The instrument ignores commands consisting of any combination of white space characters and line feeds.

Abbreviating

You can abbreviate many instrument commands. Each command in this documentation shows the abbreviations in capitals. For example, enter the command *AWGControl:RMODE TRIGgered* simply as *AWGC:RMODE TRIG*.

Concatenating

Use a semicolon (;) to concatenate any combination of set commands and queries.
The instrument executes concatenated commands in the order received. When concatenating commands and queries, follow these rules:

1. Separate completely different headers by a semicolon and by the beginning colon on all commands except the first one.
   For example, the commands *OUTPut1:STATe ON* and *AWGControl:RMODe TRIGgered*, can be concatenated into the following single command:
   OUTPut1:STATe ON;:AWGControl:RMODE TRIGgered.
2. If concatenated commands have headers that differ by only the last mnemonic, abbreviate the second command and eliminate the beginning colon.
   For example, concatenate the commands SEQuence:ELEM1:VOLTage:HIGH1 2 and SEQuence:ELEM1:VOLTage:LOW1 -2 into a single command: SEQuence:ELEM1:VOLTage:HIGH1 2;LOW1 -2
   The longer version works equally well:
   SEQuence:ELEM1:VOLTage:HIGH1 2;: SEQuence:ELEM1:VOLTage:LOW1 -2
3. Never precede a star (*) command with a semicolon (;) or colon (:).
4. When you concatenate queries, the responses to all the queries are concatenated into a single response message.
   For example, if the high level of the marker1 of channel one is 1.0 V and the voltage selection is LINE, the concatenated query : MARKer:LEVel1?; : MARKer:MODE?; will return the following:
   1.0;FIXEDLow

5. Set commands and queries may be concatenated in the same message. For example, *AWGControl:RMODe CONTinuous;:SEQuence:LENGth?* is a valid message that sets the run mode to Sequence. The message then queries the length of the sequence. Concatenated commands and queries are executed in the order received.

Here are some invalid concatenations:

- *OUTPut1:STATe ON;AWGControl:RMODE CONTinuous*
  (no colon before AWGControl)
- SEQuence:ELEM1:VOLTage:HIGH1 2;:LOW1 -2 (extra colon before LOW1 -2 instead SEQuence:ELEM1:VOLTage:HIGH1 2;LOW1 -2)

Terminating

This documentation uses <EOM> (end of message) to represent a message terminator.

Symbol	Meaning
<EOM>	Message terminator

Table 5: Message terminator and meaning

For messages sent to the instrument, the end-of-message terminator must be the END message (EOI asserted concurrently with the last data byte). The instrument always terminates messages with LF and EOI. It allows white space before the terminator. For example, it allows CR LF.

2.1.4 Parameter Types

Parameters are indicated by angle brackets, such as <file_name>. There are several different types of parameters, as listed in the following table. The parameter type is listed after the parameter. Some parameter types are defined specifically for the instrument command set and some are defined by SCPI.

Parameter type	Description	Example
Arbitrary block	A block of data bytes	#512234xxxxx... where 5 indicates that the following 5 digits (12234) specify the length of the data in bytes; xxxxx... indicates actual data or #0xxxxx...<LF><&EOI>
Boolean	Boolean numbers or values	ON or 1 OFF or 0
Discrete	A list of specific values	MINimum, MAXimum
NR1 numeric	Integers	0, 1, 15, –1
NR2 numeric	Decimal numbers	1.2, 3.141,–6.5

NR3 numeric	Floating point numbers	3.1415E+9
NRf numeric	Flexible decimal numbers that may be type NR1, NR2, or NR3	See NR1, NR2, and NR3 examples in this table
String	Alphanumeric characters (must be within quotation marks)	"Testing 1, 2, 3"

Table 6: Parameter types, their descriptions, and examples

Quoted String

Some commands accept or return data in the form of a quoted string, which is simply a group of ASCII characters enclosed by a single quote (') or double quote ("). For example: "this is a quoted string". This documentation represents these arguments as follows:

Symbol	Meaning
<QString >	Quoted string of ASCII text

Table 7: String symbol and meaning

A quoted string can include any character defined in the 7-bit ASCII character set. Follow these rules when you use quoted strings:

1. Use the same type of quote character to open and close the string. For example: "this is a valid string".
2. You can mix quotation marks within a string as long as you follow the previous rule. For example, "this is an 'acceptable' string".
3. You can include a quote character within a string simply by repeating the quote. For example: "here is a "" mark".
4. Strings can have upper or lower case characters.
5. A carriage return or line feed embedded in a quoted string does not terminate the string, but is treated as just another character in the string.
6. The maximum length of a quoted string returned from a query is 1000 characters.

Here are some invalid strings:

- "Invalid string argument' (quotes are not of the same type)
- "test<EOI>" (termination character is embedded in the string)

Units and SI Prefix

If the decimal numeric argument refers to voltage, frequency, impedance, or time, express it using SI units instead of using the scaled explicit point input value format <NR3>. (SI units are units that conform to the System International d'Unites standard.) For example, use the input format 200 mV or 1.0 MHz instead of 200.0E-3 or 1.0E+6, respectively, to specify voltage or frequency.

Omit the unit when you describe commands, but include the SI unit prefix. Enter both uppercase and lowercase characters. The following list shows examples of units you can use with the commands.

- V for voltage (V).
- HZ for frequency (Hz).
- OHM for impedance (ohm).
- S for time (s).
- DBM for power ratio.
- PCT for %.
- VPP for Peak-to-Peak Voltage (V p-p).
- UIPP for Peak-to-Peak, Unit is UI (UI p-p).
- UIRMS for RMS, Unit is UI (UIrms).
- SPP for Peak-to-Peak, Unit is second (s p-p).
- SRMS for RMS, Unit is second (srms).
- V/NS for SLEW's unit (V/ns).

In the case of angles, use RADian and DEGree. The default unit is RADian. The SI prefixes, which must be included, are shown in the following table. You can enter both uppercase and lowercase characters.

SI prefix [1]	Corresponding power
EX	10^{18}
PE	10^{15}
T	10^{12}
G	10^{9}
MA	10^{6}
K	10^{3}
M	10^{-3}
U [2]	10^{-6}
N	10^{-9}
P	10^{-12}
F	10^{-15}
A	10^{-18}

Table 8: SI prefixes and their indexes

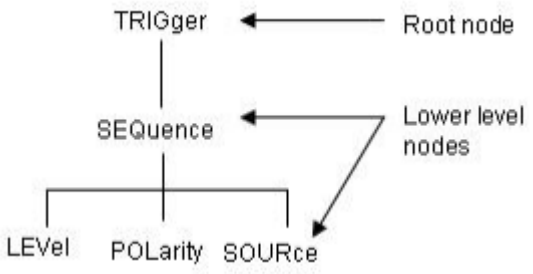
1. Note that the prefix m/M indicates 10^{-3} when the decimal numeric argument denotes voltage or time, but indicates 10^{6} when it denotes frequency.
2. Note that the prefix u/U is used instead of "µ".

Since M (m) can be interpreted as 1E-3 or 1E6 depending on the units, use mV for V, and MHz for Hz.

The SI prefixes need units.
Correct: 10MHz, 10E+6Hz, 10E+6
Incorrect: 10M

2.1.5 SCPI Commands and Queries

The arbitrary waveform generator uses a command language based on the SCPI standard. The SCPI (Standard Commands for Programmable Instruments) standard was created by a consortium to provide guidelines for remote programming of instruments. These guidelines provide a consistent programming environment for instrument control and data transfer. This environment uses defined programming messages, instrument responses and data formats that operate across all SCPI instruments, regardless of manufacturer.
The SCPI language is based on a hierarchical or tree structure that represents a subsystem (see following figure). The top level of the tree is the root node; it is followed by one or more lower-level nodes.



You can create commands and queries from these subsystem hierarchy trees.
Commands specify actions for the instrument to perform. Queries return measurement data and information about parameter settings.

2.2 Status and events

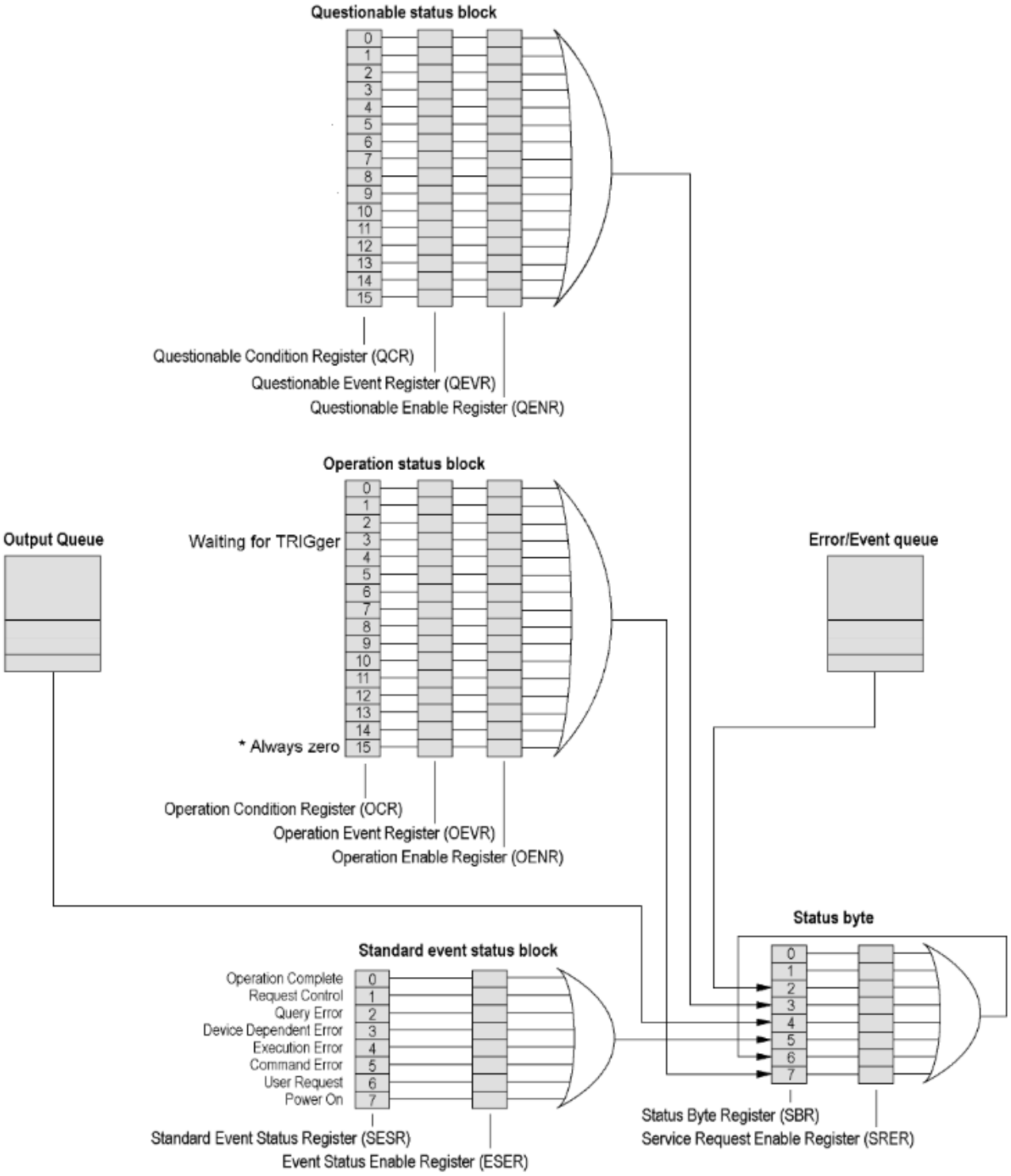
The SCPI interface in the instrument includes a status and event reporting system that enables the user to monitor crucial events that occur in the instrument.

2.2.1 Status and event reporting system

The following figure outlines the status and event reporting mechanism offered in the arbitrary waveform generators. It contains three major blocks:

- Standard Event Status
- Operation Status
- Questionable Status (fan-out structure, not used in this version).

The processes performed in these blocks are summarized in the Status Byte.


Questionable status block
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
Questionable Condition Register (QCR)
Questionable Event Register (QEVR)
Questionable Enable Register (QENR)
Operation status block
Output Queue
Waiting for TRIGger
Error/Event queue
* Always zero
Operation Condition Register (OCR)
Operation Event Register (OEVR)
Operation Enable Register (OENR)
Status byte
Standard event status block
Operation Complete
Request Control
Query Error
Device Dependent Error
Execution Error
Command Error
User Request
Power On
Status Byte Register (SBR)
Service Request Enable Register (SRER)
Standard Event Status Register (SESR)
Event Status Enable Register (ESER)

2.2.2 Status Byte Register (SBR)

The bits of this register are used to monitor the output queue, SESR and service requests, respectively. The contents of this register are returned when the *STB? query is used.

BIT	Name	Description
7 (MSB)	OSS	Operation Summary Status (OSS). Summary of the operation status register.
6	RQS/MSS	Request Service (RQS)/Master Status Summary (MSS). When the instrument is accessed using the serial poll command, this bit is called the Request Service(RQS) bit and indicates to the controller that a service request has occurred. The RQS bit is cleared when serial poll ends. When the instrument is accessed using the *STB? query, this bit is called the Master Status Summary (MSS) bit and indicates that the instrument has issued a service request for one or more reasons. The MSS bit is never cleared to 0 by the *STB? Query.
5	ESB	Event Status Bit (ESB). This bit indicates whether or not a new event has occurred after the previous Standard Event Status Register (SESR) has been cleared or after an event readout has been performed
4	MAV	Message Available Bit (MAV). This bit indicates that a message has been placed in the output queue and can be retrieved.
3	QSS	Questionable Summary Status (QSS). Summary of the Questionable Status Byte register.
2	EAV	Event Quantity Available (EAV). Summary of the Error Event Queue.
1	--	This is not used
0(LSB)	--	This is not used

Table 9: Status Byte Register (SBR)

2.2.3 Service Request Enable Register (SRER)

The SRER is made up of bits defined exactly the same as bits 0 through 7 in the SBR as shown in the following figure. This register is used by the user to determine what events will generate service requests.
The SRER bit 6 cannot be set. Also, the RQS is not maskable.
The generation of a service request with the GPIB interface involves changing the SRQ line to LOW and making a service request to the controller. The result is that a status byte for which an RQS has been set is returned in response to serial polling by the controller.
Use the *SRE command to set the bits of the SRER. Use the *SRE? query to read the contents of the SRER. Bit 6 must normally be set to 0.

7	6	5	4	3	2	1	0
OSB	--	ESB	MAV	QSB	--	--	--

Table 10: Service Request Enable Register (SRER)

2.2.4 Standard Event Status Block (SESB)

Reports the power on/off state, command errors, and the running state. It consists of the following registers:

- Standard Event Status Register (SESR)
- Event Status Enable Register (ESER)

These registers are made up of the same bits defined in the following table. Use the *ESR? query to read the contents of the SESR. Use the *ESE() command to access the ESER

2.2.4.1 Standard Event Status Register (SESR)

BIT	Name	Description
7 (MSB)	PON	Power On (PON). Indicates that the power to the instrument is on.
6	--	Not used.
5	CME	Command Error (CME). Indicates that a command error has occurred while parsing by the command parser was in progress.

BIT	Name	Description
4	EXE	Execution Error (EXE). Indicates that an error occurred during the execution of a command. Execution errors occur for one of the following reasons: • when a value designated in the argument is outside the allowable range of the instrument, or is in conflict with the capabilities of the instrument. • when the command could not be executed properly because the conditions for execution differed from those essentially required.
3	DDE	Device-Dependent Error (DDE). An instrument error has been detected.
2	QYE	Query Error (QYE). Indicates that a query error has been detected by the output queue controller. Query errors occur for one of the following reasons: • an attempt was made to retrieve messages from the output queue, despite the fact that the output queue is empty or in pending status. • the output queue messages have been cleared despite the fact that they have not been retrieved.
1	--	Not used.
0 (LSB)	OPC	Operation Complete (OPC). This bit is set with the results of the execution of the *OPC command. It indicates that all pending operations have been completed.

Table 11: Standard Event Status Register (SESR)

When an event occurs, the SESR bit corresponding to the event is set, resulting in the event being stacked in the Error/Event Queue. The SBR OAV bit is also set. If the bit corresponding to the event has also been set in the ESER, the SBR ESB bit is also set. When a message is sent to the Output Queue, the SBR MAV bit is set.

2.2.5 Operation status block

The operation status block contains conditions that are part of the instrument's normal operation. It consists of the following registers:

- Operation Condition Register (OCR)

- Operation Event Register (OEVR)
- Operation Enable Register (OENR)

These registers are made up of the same bits defined in the following table. Use the STATus:OPERation commands to access the operation status register set.

2.2.5.1 Operation Condition Register (OCR)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
--	--	--	--	--	--	--	--	--	--	--	--	WFT	--	--	--

Table 12: Operation Condition Register (OCR)

Bit3 Waiting for trigger (WFT): indicates that the instrument is waiting for a trigger event to occur.

When the specified state changes in the OCR, its bit is set or reset. This change is filtered with a transition register, and the corresponding bit of the OEVR is set.
If the bit corresponding to the event has also been set in the OENR, the SBR OSS bit is also set.

2.2.6 Questionable status block

The questionable status register set contains bits which give an indication of the quality of various aspects of the signal together with the fanned out registers as described in the next subsections. It consists of the following registers:

- Questionable Condition Register (QCR)
- Questionable Event Register (QEVR)
- Questionable Enable Register (QENR)

Despite the commands to query these registers are usable (see STATus:QUEStionable commands) , these register are not used in this version.

2.3 Analog data format (.txt file only)

The analog waveform can be imported into the instrument using a .txt file. For analog waveform you have to create a single column of values (signed integer or signed decimal, the header is not allowed) separated with 'new line'. These values are imported into the instrument normalized so that the user can easily adjust waveform's amplitude/offset using the corresponding SCPI commands (see SEQuence:ELEM[n]:AMPlitude[m], SEQuence:ELEM[n]:OFFset[m] or SEQuence:ELEM[n]:VOLTage:HIGH[m], SEQuence:ELEM[n]:VOLTage:LOW[m] commands).

2.4 Digital Data format (.txt file only)

For digital waveform you have to create a single column of values (unsigned integer range [0..(2^{32} - 1)], the header is not allowed) separated with 'new line'. In this way each value converted into 32-bits binary format represents the status of the corresponding digital line (Bit 0 -> Digital Line 0, Bit 1 -> Digital Line 1, …, Bit 31 -> Digital Line 31).
For example the integer value 5789 matches the binary value 00000000000000000001011010011101, thereby the status of digital Pads is:

Digital Pod A

1	0	0	1	1	1	0	1
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Digital Pod B

0	0	0	1	0	1	1	0
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Digital Pod C

0	0	0	0	0	0	0	0
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Digital Pod D

0	0	0	0	0	0	0	0
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

If the configuration of the instrument implies that some pods are not present or enabled the corresponding bits are meaningless.

The digital outputs sampling rate is the same of the analog sampling rate, so the length of the digital samples must be the same of the analog waveform length.
Example: analog waveform length = 100 samples
→ digital waveform length = 100 samples

2.5 Granularity

The minimum waveform length is 32 samples.
From 32 to 384 samples the granularity is 16.
With waveform's length greater than 384 the granularity is 1.

2.6 Transferring Data File

When transferring data file, it is convenient to send data in chunks. This allows better memory management and enables you to stop the transfer before it is completed. It also helps the external controller to report the progress of the operation to the user.

2.6.1 Block Data Format

Block data is a transmission format which is suitable for the transmission of large amounts of data. A command using a block data parameter with definite length has the following structure:

Example: HEADer:HEADer #45168xxxxxxxx

The hash symbol # introduces the data block. The next number indicates how many of the following digits describe the length of the data block. In the example the 4 following digits indicate the length to be 5168 bytes. The data bytes follow. During the transmission of these data bytes all End or other control signs are ignored until all bytes are transmitted.

2.7 Byte Order During Transfer

Waveform data is always transferred using little-endian format.

2.8 How to generate an arbitrary waveform

If you need to generate an arbitrary waveform and reproduce it on you instrument, you can follow the instructions below:

1. Generate a .txt file with the waveform samples; fill the .txt file with all arbitrary samples paying attention to data format (see analog/digital data format chapters for more details).

2. Transfer the .txt file from client PC to mass storage memory of the instrument (see MMEMory:DATA command and Block Data Format chapter for more details about this operation).

3. Import the .txt file in waveform list (see WLISt:WAVeform:IMPort command which also allows you to choose an appropriate name for the new imported waveform, i.e. "Wave_name") .

4. Insert "Wave_name" waveform in the first entry of sequencer and on the desired channel (see SEQuence:ELEM[n]:WAVeform[m] command).

5. Set the desired running mode (AWGControl:RMODe) and output parameters.

6. Enable the outputs (OUTPut[n][:STATe] or DIGitals:STATe).

7. Send a start command (AWGControl:RUN).

2.9 Command Groups

The following commands refer to the parameters [n] and [m] that depend on the instrument model.

Instrument Model	Parameter [n] = Available Channels	Parameter [m] = Available Marker Outputs
Model 675-2C	1 | 2	1
Model 675-4C	1 | 2 | 3 | 4	1 | 2
Model 675-8C	1 | 2 | 3 | 4 | 5 | 6 | 7 | 8	1 | 2 | 3 | 4

Table 13: Models and available parameters

2.9.1 Control group commands

Use the following commands to control operating modes:

Command	Description
AWGControl:AFGSwitch	Turns off the TrueArb application and runs the Simple AFG application
AWGControl:BURST	Sets or returns the Burst Count parameter
AWGControl:CONFigure:CNUMber	Returns the number of analog channels available on the instrument
AWGControl:CONFigure:DNUMber	Returns the number of digital channels available on the AWG
AWGControl:DECreasing	Sets or returns the Sample Drecreasing Strategy parameter
AWGControl:INCreasing	Sets or returns the Sample Increasing Strategy parameter
AWGControl:LENGth:MODE	Sets or returns the Entry Length Strategy parameter
AWGControl:RESET[:IMMediate]	This command resets sequence to its default state.
AWGControl:RMODe	This command sets or returns the AWG run mode.
AWGControl:RSTATe	Returns the state of the arbitrary waveform generator.
AWGControl:RUN[:IMMediate]	Initiates the output of a waveform or a sequence
AWGControl:SREStore	Opens a setup file into the AWG's setup memory

Command	Description
AWGControl:SSAVe	Saves the AWG's setup with waveforms
AWGControl:STOP[:IMMediate]	Stops the output of a Sequence
AWGControl:WAITstate	Sets or returns the Wait Trigger On parameter
AWGControl:JUMPMode	Sets or returns the Jump Mode parametes
AWGControl:DJStrobe	Sends the pattern strobe event

Table 14: Control group commands

2.9.2 Calibration and Diagnostic

Command	Description
CALibration[:ALL]	Performs a full calibration of the AWG. The query form performs a full calibration and returns a status of the operation.
DIAGnostic[:ALL]	Performs the self diagnostic procedure.

Table 15: Calibration and Diagnostic group commands

2.9.3 Output Group Commands

Use the following output commands to set or return the characteristics of the output port of the arbitrary waveform generator:

Command	Description
OUTPut[n]:BLOFfset	Sets or returns the Base Line Offset parameter of the channel n.
OUTPut[n]:POLarity	Sets or returns the Polarity parameter of the channel n
OUTPut[n]:SCALe	Sets or returns the Amplitude Scale parameter of the channel n
OUTPut[n]:SERIESIMPedance	Sets or returns the Output Impedance parameter for the channel n.
OUTPut[n][:STATe]	Sets or returns the channel n state.
DIGitals:LEVel[m]	Sets or returns the Voltage Level of the selected Digital Probe "m"

DIGitals:NUMber	Sets or returns the number of the Digital Channels.
DIGitals:SKEW[m]	Sets or returns the Skew parameter for the selected Digital Probe “m”
DIGitals:STATe	Sets or returns the state of the dgital channels.

Table 16: Output group commands

2.9.4 Display Commands

Display commands let you to manage features related to the user interface.

Command	Description
DISPlay:FOCus	Selects the channel displayed "in front" on the AWG
DISPlay:UNIT:VOLT	Selects the method for specifying voltage ranges
DISPlay[:WINDow]:TEXT:CLEar	Delete text message
DISPlay[:WINDow]:TEXT[:DATA]	Set or query the text message display
HCOPy:SDUMp[:IMMediate]	Copy screen image and save it in the specified file.

Table 17: Display group commands

2.9.5 License Commands

License commands let you to manage features related to the options that can be installed through a license file.

Command	Description
LICense:ERRor	This query-only command returns a code about license options loading operation.
LICense:HID	Returns the instrument HostID unique identifier.
LICense:INSTall	Accepts a license and installs it on the instrument.
LICense:LIST	Returns the license codes as a comma-separated list of string.
*OPT	Returns the implemented options for the AWG.

Table 18: License group commands

2.9.6 Marker Commands

Use the following marker commands to set and query the marker output parameter:

Command	Description
MARKer:LEVel[m]	Sets or returns the Voltage Level parameter of the Marker "m"
MARKer:MODE[m]	Sets or returns the Marker Mode parameter of the Marker "m"
MARKer:SKEW[m]	Sets or returns the Marker Skew parameter of the Marker "m"

Table 19: Marker group commands

2.9.7 Clock Group Commands

Use the following commands to set and query the reference and sampling clock parameters

Command	Description
ROSCillator	Sets or returns the reference clock value in Hz
ROSCillator:SOURce	Sets or returns the reference clock source to internal or external
AWGControl:SRATe	This command sets or returns the sample rate for the clock

Table 20: Clock group commands

2.9.8 IEEE Mandated and Optional Group Command

All AWG IEEE mandated and optional command implementations are based on the SCPI standard and the specifications for devices in IEEE 488.2.

Command	Description
*CAL	Runs the self calibration and returns the result. Same as CALibration[:ALL].
*CLS	Clears all event registers and queues.
*ESE	This command sets or returns the status of Event Status Enable Register (ESER).

	(See Status and events chapter).
*IDN	This command returns identification information for the AWG. Refer to Std IEEE 488.2 for additional information.
*OPC	This command causes the AWG to sense the internal flag referred to as the "No-Operation-Pending" flag. The command sets bit 0 in the Standard Event Status Register when pending operations are complete. The query form returns a "1" when the last overlapping command operation is finished.
*RST	Resets the AWG to its default state.
*SRE	Sets or returns the bits in the Service Request Enable Register (SRER).
*TRG	This command generates a trigger event. This is equivalent to press and release the trigger button on the front panel.
*TST	Executes ths Self Diagnostic Procedure.
*WAI	Ensures the completion of the previous command before the next command is issued.

Table 21: IEEE Mandatory group commands

2.9.9 Status Group Command

The external controller uses the status commands to coordinate operation between the TrueArb and other devices on the bus. The status commands set and query the registers/queues of the TrueArb event/status reporting system. For more information about registers and queues, see Status and Event reporting section.

Command	Description
*CLS	Clears all event registers and queues
*ESE	Sets or queries the status of Event Status Enable Register (ESER)
*ESR	Returns the status of Standard Event Status Register (SESR)
*SRE	Sets or queries the bits in Service Request Enable Register (SRER)
STATus:OPERation:CONDition	Returns the contents of the Operation Condition Register (OCR)

STATus:OPERation:ENABle	Sets or returns the mask for the Operation Enable Register (OENR)
STATus:OPERation[:EVENt]	Returns the contents of Operation Event Register (OEVR)
STATus:PRESet	Sets the OENR and QENR registers.
STATus:QUEStionable:CONDition	Returns the status of the Questionable Condition Register (QCR)
STATus:QUEStionable:ENABle	Sets or returns the mask for Questionable Enable Register (QENR).
STATus:QUEStionable[:EVENt]	Returns the status of the Questionable Event (QEVR) Register and clears it– Not used.
*STB	Returns the contents of Status Byte Register (SBR).
*PSC	Set or query power-on status clear

Table 22:Status group commands

2.9.10 System Group Commands

Use the following system commands to control miscellaneous instrument functions:

Command	Description
SYSTem:BEEPer:STATe	Sets or queries the beeper state
SYSTem:BEEPer[:IMMediate]	Generates an audible tone
SYSTem:DATE	Sets or returns the system date.
SYSTem:ERRor[:NEXT]	Returns data from the error and event queue.
SYSTem:KLOCk[:STATe]	Sets or queries the front panel lock/unlock
SYSTem:SECurity:IMMediate	Resets to factory default
SYSTem:TIME	Sets or returns the system time.
SYSTem:VERSion	Returns the SCPI version number to which the command conforms.

Table 23: System group commands

2.9.11 Memory Group Commands

Memory commands let you manage the setup memory. The following table describes the memory commands.

Command	Description
*RCL	Recall instrument settings from setup memory
*SAV	Save instrument settings to setup memory

Command	Description
MEMory:NSTates	Returns the total number of available configurations saved in the AWG.
MEMory:STATe:CATalog	List the names of available configurations saved in the AWG.
MEMory:STATe:DELete	Delete a configuration saved in the AWG.
MEMory:STATe:DELete	Delete the setup memory
MEMory:STATe:LOCK	Sets or queries the lock of configuration overwrite and deletion
MEMory:STATe:NAME	Copies a configuration.
MEMory:STATe:VALid	Queries the availability of a configuration.
DELete:SETUp	Deletes a configuration.
RECALL:SETUp	Restores the instrument settings from a configuration name.

Table 24: Memory Group commands

2.9.12 MASS Memory Commands

Mass memory commands let you change mass memory attributes. The following table describes the mass memory commands.

Command	Description
MMEMory:CATalog	Returns the entire list of files and directory located in the current directory.
MMEMory:CDIRectory	Sets or returns the current directory of the file system on the AWG.
MMEMory:COPY	Copies a source file in a target file.
MMEMory:DATA	Sets or returns block data to/from file in the current mass storage device.
MMEMory:DATA:SIZE	Returns the size in bytes of a selected file.
MMEMory:DELete	Deletes a file or directory from the AWG's files system.
MMEMory:DOWNload:DATA	Downloads data from the host computer to instrument's Mass Memory.
MMEMory:DOWNload:FNAMe	Specifies file name for downloading data from the computer to instrument's Mass Memory.
MMEMory:EXPort	Exports a waveform from the current waveform list to an archive file (.zip).

Command	Description
MMEMory:IMPort	Imports a file into the AWG's waveform list.
MMEMory:LOAD:ALL	Loads an AWG's configuration file and set it as current configuration.
MMEMory:LOAD:STATe	Loads an AWG's configuration file in the configurations list.
MMEMory:MDIRectory	Creates a new directory in the current path on the Mass Memory system.
MMEMory:MOVE	Moves a file on Mass Memory device.
MMEMory:MSIS	Sets or returns a mass storage device used by all MMEMory commands.
MMEMory:OPEN	Loads a file into the AWG waveform list.
MMEMory:OPEN:SETup	Loads an AWG's configuration file and set it as current configuration.
MMEMory:RDIRectory	Removes an empty directory.
MMEMory:SAVE:SETup	Saves the current configuration in an archive (.zip).
MMEMory:STORe:ALL	Saves the current configuration in an archive (.zip).
MMEMory:STORe:STATe	Saves a configuration present in the configurations list in an archive (.zip)
MMEMory:UPLoad	Returns the contents of a file.

Table 25: Mass Memory Group commands

2.9.13 Trigger Group Commands

The trigger commands let you control all aspects of triggering. The following table describes the trigger input commands.

Command	Description
ABORt	Reset and initialize the trigger system
TRIGger[:SEQuence][:IMMediate]	Generates a trigger event
TRIGger[:SEQuence]:SLOPe	Set or query the slope of the trigger input signal
TRIGger[:SEQuence]:SOURce	Set or query the source of the trigger input signal
TRIGger[:SEQuence]:LEVel	Set or query the trigger threshold level of an input signal
TRIGger[:SEQuence]:TIMer	Set or query the internal rate of the timer
TRIGger:IMPedance	Set or query the trigger input impedance

Table 26: Trigger group commands

2.9.14 Sequence Group Commands

The following set of commands provides ways to create and edit the waveform sequences in the instruments. When the instrument runs a sequence, it outputs the waveforms in the order defined in the sequence.

Important Note: there is only one sequence defined for an instrument. For each entry of the sequencer the number of repetitions and waveform's length are common to all channels, while Amplitude/Offset (Voltage High/Low) and waveform' shape are independent of each other.

Use the following sequence commands to define and edit a sequence:

Command	Description
SEQuence:ELEM[n]:AMPlitude[m]	Sets or returns peak to peak voltage level for sequence element n of channel m.
SEQuence:ELEM[n]:OFFset[m]	Sets or returns the offset for sequence element n of channel m.
SEQuence:ELEM[n]:VOLTage:HIGH[m]	Sets or returns the maximum level of the waveform expressed in Volts for sequence element n of channel m.
SEQuence:ELEM[n]:VOLTage:LOW[m]	Sets or returns the minimum level of the waveform expressed in Volts for sequence element n of channel m.
SEQuence:ELEM[n]:LENGth	Sets or returns the number of waveform samples for sequence element n.
SEQuence:ELEM[n]:LOOP:COUNt	Sets or returns the number of repetitions for sequence element n.
SEQuence:ELEM[n]:WAVeform[m]	Sets or returns the waveform (among those in the waveforms list) to be assigned to the sequence element n of channel m.
SEQuence:LENgth	Sets or returns the number of elements in the sequencer.
SEQuence:NEW	Creates a new sequence.
SEQuence:FOCus	Sets which sequence element is shown on the display.
SEQuence:ELEM[n]:WAITEvent	Sets or returns the wait event type for sequence element n.
SEQuence:ELEM[n]:GOTOMode	Sets or returns the "Go To" command type for sequence element n

SEQuence:ELEM[n]:GOTOEntry	Sets or returns the target entry for the "GOTO" command for the sequence element 'n'
SEQuence:ELEM[n]:JUMPTOMode	Sets or returns the "Jump To" command type for sequence element n
SEQuence:ELEM[n]:JUMPEvent	Sets or returns the jump event type for sequence element n.
SEQuence:ELEM[n]:JUMPTOEntry	Sets or returns the target entry index for the "Jump To" command.
SEQuence:ELEM[n]:PATTERN	Sets or returns the pattern code value for the "Pattern Jump" command for sequence element n.
SEQuence:ELEM[n]:PATTERNJUMPTOMode	Sets or returns the "Pattern Jump" command type for sequence element n.
SEQuence:ELEM[n]:PATTERNJUMPTOEntry	Sets or returns the target entry for the "Pattern Jump" command for the sequence element 'n'

Table 27: Sequence group commands

2.9.15 Waveform Group Commands

Use the following waveform commands to create and transfer waveforms between the instrument and the external controller:

Command	Description
WLISt:LIST	Returns a list of all waveform names in the waveform list.
WLISt:NAME	Returns the waveform name of an element which is in a specific position in the waveform list.
WLISt:SIZE	Returns the size of the waveform list.
WLISt:WAVeform:DATA	Transfers waveform data of a waveform in waveform list to the external control program.
WLISt:WAVeform:DELete	Deletes a waveform from the waveform list or all imported waveforms.
WLISt:WAVeform:IMPort	Imports a waveform from internal driver or USB driver into the waveform list.
WLISt:WAVeform:LENGth	Returns the size of the specified waveform in the waveform list.
WLISt:WAVeform:LMAXimum	Returns the maximum number of waveform sample points allowed.

Command	Description
WLISt:WAVeform:LMINimum	Returns the minimum number of waveform sample points required for a valid waveform.
WLISt:WAVeform:PREDefined	Returns true or false based on whether the waveform is predefined (already present in waveform list by default).
WLISt:WAVeform:TYPE	Returns the type of the waveform (analog or digital).

Table 28: Waveform group commands

2.9.16 Multi Instrument Commands

Use the following commands to synchronize multiple instrument.
The multi instrument synchronization is available on 8 channel models only.

Command	Description
MIM:CAPTure	This command captures all slave instruments
MIM:ID	This command returns the identification number of the device in the multi-instrument chain.
MIM:CAPTured	Returns whether the instrument has been captured by a master.
MIM:FORWard	Returns whether there is another instrument connected to the "Sync Out" port.
MIM:SLAve	Returns whether there is another instrument connected to the "Sync In" port.
MIM:NUMber	Returns the number of captured devices.
MIM:RELease	Releases all the captured instruments.

Table 29: Multi-Instrument group commands and their descriptions

2.10 Control Group Commands

Command	AWGControl:AFGSwitch (No Query Form)
Description	This command turns off the TrueArb application and runs the Simple AFG application
Group	Control
Syntax	AWGControl:AFGSwitch
Related Commands	AWGControl:APPSwitch
Arguments	None
Returns	None
Example	AWGControl:AFGSwitch Runs the AFG application

Table 30: AWGControl:AFGSwitch

Command	AWGControl:BURST
Description	This command and query set or return the Burst Count parameter
Group	Control
Syntax	AWGControl:BURST {MINimum | MAXimum | DEFault | <value>} AWGControl:BURST? [{MINimum | MAXimum}]
Related Commands	None
Arguments	• MINimum: sets the parameter to the minimum value • MAXimum: sets the parameter to the maximum value • DEFault: sets the parameter to the default value • <value> := <NR1> is the burst count
Returns	A single <NR1> value
Example	AWGControl:BURST DEFault It sets the Burst Count parameter to its default value AWGControl:BURST 20 AWGControl:BURST? Might return 20

Table 31: AWGControl:BURST

Command	AWGControl:CONFigure:CNUMber (Query Only)
Description	This command returns the number of analog channels available on the AWG.
Group	Control
Syntax	AWGControl:CONFigure:CNUMber?
Related Commands	

Arguments	
Returns	A single <NR1> value
Example	AWGControl:CONFigure:CNUMber? Might return 2

Table 31: AWGControl:CONFigure:CNUMber

Command	AWGControl:CONFigure:DNUMber (Query Only)
Description	This command returns the number of digital channels available on the AWG.
Group	Control
Syntax	AWGControl:CONFigure:DNUMber?
Related Commands	
Arguments	
Returns	A single <NR1> value
Example	AWGControl:CONFigure:DNUMber? Might return 8

Table 32: AWGControl:CONFigure:DNUMber

Command	AWGControl:DECreasing
Description	This command and query sets or returns the Sample Decreasing Strategy. The "Sample decreasing strategy" parameter defines the strategy used to adapt the waveform length to the sequencer entry length in the case where the original waveform length is longer than the sequencer entry length
Group	Control
Syntax	AWGControl:DECreasing {DECIMation | CUTTail | CUTHead} AWGControl:DECreasing?
Related Commands	
Arguments	• DECIMation: it reduces the number of samples maintaining the waveform shape • CUTTail: it cuts the tail of the waveform reducing its size • CUTHead: it cuts the head of the waveform reducing its size.
Returns	DECIMation | CUTTail | CUTHead

Example	AWGControl:DECreasing CUTTail sets the decreasing strategy to cut tail AWGControl:DECreasing? Might return DECIMation indicating that the decreasing strategy is set to decimation

Table 33: AWGControl:DECreasing

Command	AWGControl:INCreasing
Description	This command and query sets or returns the Sample Increasing Strategy. The "Sample increasing strategy" parameter defines the strategy used to adapt the waveform length to the sequencer entry length in the case where the original waveform length is shorter than the sequencer entry length.
Group	Control
Syntax	AWGControl:INCreasing {INTERpolation | RETURNzero | HOLDlast | SAMPLESMultiplication} AWGControl:INCreasing?
Related Commands	
Arguments	INTERpolation: it performs a linear interpolation between the waveform samples RETURNzero: it fills with '0's the tail of the waveform HOLDlast: it holds the last value of the waveform SAMPLESMultiplication: it repeats the waveform samples
Returns	INTERpolation | RETURNzero | HOLDlast | SAMPLESMultiplication
Example	AWGControl: INCreasing INTERpolation sets the increasing strategy to interpolation. AWGControl: INCreasing? Might return HOLDlast indicating that the Sample increasing strat-egy is seto to Hold Last.

Table 34: AWGControl:INCreasing

Command	AWGControl:LENGth:MODE
Description	This command and query sets or returns the Entry Length Strategy. This strategy manages the length of the sequencer entries in relationship with the length of the channel waveforms defined for each entry

Group	Control
Syntax	AWGControl:LENGth:MODE {ADAPTLonger | ADAPTShorter | DEFault} AWGControl:LENGth:MODE?
Related Commands	
Arguments	• ADAPTLonger: the length of an entry of the sequencer by default will be equal to the length of the longer channel waveform, among all analog channels, assigned to the entry. • ADAPTShorter: the length of an entry of the sequencer by default will be equal to the length of the shorter channel waveform, among all analog channels, assigned to the entry. • DEFault:the length of an entry of the sequencer by default will be equal to the value specified in the Sequencer Item Default Length [N] parameter
Returns	ADAPTLonger | ADAPTShorter | DEFault
Example	AWGControl:LENGth:MODE ADAPTLonger sets the entry length strategy to "Adapt to Longer Analog Waveform" AWGControl:LENGth:MODE? Might return DEFault

Table 35: AWGControl:LENGth:MODE

Command	AWGControl:RESET[:IMMediate] (No Query Form)
Description	This command resets sequence to its default state. The sequence table will be cleared. Parameters such as sampling rate, skew time will be set to its default value. **Importante Note**: the behaviour of this command is equivalent to *RST with the exception that before sending the AWGControl:RESET command, the AWG must be in Stopped state.
Group	Control
Syntax	AWGControl:RESET[:IMMediate]
Related Commands	
Arguments	
Returns	None
Example	AWGControl:RESET If the AWG is Stopped, it will reset the AWG to its default state.

Table 36: AWGControl:RESET[:IMMediate]

Command	AWGControl:RMODe {CONTinuous | BURSt | TCONtinuous | STEPped | ADVAnced }

Description	This command sets or returns the AWG run mode.
Group	Control
Syntax	AWGControl:RMODe AWGControl:RMODe?
Related Commands	
Arguments	• CONTinuous: each waveform will loop as written in the entry repetition parameter and the entire sequence is repeated circularly • BURSt: the AWG waits for a trigger event. When the trigger event occurs each waveform will loop as written in the entry repetition parameter and the entire sequence will be repeated circularly many times as written in the Burst Count[N] parameter. If you set Burst Count[N]=1 the instrument is in Single mode and the sequence will be repeated only once. • TCONtinuous: the AWG waits for a trigger event. When the trigger event occurs each waveform will loop as written in the entry repetition parameter and the entire sequence will be repeated circularly. • STEPped: the AWG, for each entry, waits for a trigger event before the execution of the sequencer entry. The waveform of the entry will loop as written in the entry repetition parameter. After the generation of an entry has completed, the last sample of the current entry or the first sample of the next entry is held until the next trigger is received. At the end of the entire sequence the execution will restart from the first entry. • ADVAnced: it enables the "Advanced" mode. In this mode the execution of the sequence can be changed by using conditional and unconditional jumps (JUMPTO and GOTO commands) and dynamic jumps (PATTERN JUMP commands). The *RST command sets this parameter to CONTinuous.
Returns	CONTinuous | BURSt | TCONtinuous | STEPped |ADVAnced
Example	AWGCONTROL:RMODE STEPped sets the AWG run mode to Stepped. AWGCONTROL:RMODE? might return CONTinuous, indicating that the AWG run mode is set to Continuous.

Table 37: AWGControl:RMODe

Command	AWGControl: RSTATe (Query Only)
Description	This command returns the run state of the AWG.
Group	Control
Syntax	AWGControl:RSTATe?
Related Commands	AWGControl:RMODe
Arguments	
Returns	A single <NR1> value. 0 indicates that the AWG has stopped. 1 indicates that the AWG is waiting for trigger. 2 indicates that the AWG is running.
Example	AWGCONTROL:RSTATe? might return 0, indicating that waveform generation is stopped.

Table 38: AWGControl:RSTATe

Command	AWGControl:RUN[:IMMediate] (No Query Form)
Description	This command starts the output of a sequence. This is the same to press the run button on the front-panel or display.
Group	Control
Syntax	AWGControl:RUN[:IMMediate]
Related Commands	AWGControl:STOP[:IMMediate]
Arguments	
Returns	A single <NR1> value. 0 indicates that the AWG has stopped. 1 indicates that the AWG is waiting for trigger. 2 indicates that the AWG is running.
Example	AWGControl:RUN puts the AWG in the run state

Table 39: AWGControl:RUN[:IMMediate]

Command	AWGControl:SREStore (No Query Form)
Description	This command opens a setup file into the AWG's setup memory.
Group	Control
Syntax	AWGControl:SREStore <cfg_name>
Related Commands	MMEMory:OPEN:SETup
Arguments	<cfg_name>::=<string>
Returns	

Example	AWGControl:SREStore "my_configuration"

Table 40: AWGControl:SREStore

Command	AWGControl:SSAVe (No Query Form)
Description	This command saves the AWG setup with waveforms.
Group	Control
Syntax	AWGControl:SSAVe <cfg_name>
Related Commands	MMEMory:SAVE:SETup
Arguments	<cfg_name>::=<string>
Returns	
Example	AWGCONTROL:SSAVE "my_configuration"

Table 41: AWGControl:SSAVe

Command	AWGControl:STOP[:IMMediate] (No Query Form)
Description	This command stops the output of a sequence.
Group	Control
Syntax	AWGControl:STOP[:IMMediate]
Related Commands	AWGControl:RUN[:IMMediate]
Arguments	
Returns	
Example	AWGControl:STOP:IMMediate stops the output of a sequence.

Table 42: AWGControl:STOP[:IMMediate]

Command	AWGControl:WAITstate
Description	This command sets or returns the Wait Trigger On parameters.This parameter defines the behaviour of the output during the wait trigger condition in the Triggered Run Modes.
Group	Control
Syntax	AWGControl:WAITstate {FIRSTsample | LASTsample} AWGControl:WAITstate?
Related Commands	
Arguments	• FIRSTsample: the first waveform sample of the next entry is held until the next trigger is received • LASTsample: the last waveform sample of the current entry is held until the next trigger is received
Returns	FIRSTsample | LASTsample

Example	AWGControl:WAITstate FIRSTsample Sets the Wait Trigger On parameter to the First waveform sample AWGControl:WAITstate? Might return FIRSTsample

Table 43: AWGControl:WAITstate

Command	AWGControl:JUMPMode
Description	This command sets or returns the Jump Mode parameter.
Group	Control
Syntax	AWGControl:JUMPMode {AFTERrepetitions|IMMediate} AWGControl:JUMPMode?
Related Commands	SEQuence:ELEM[n]:JUMPTOMode SEQuence:ELEM[n]:JUMPEvent SEQuence:ELEM[n]:JUMPTOEntry
Arguments	• AFTERrepetitions: the sequencer jumps to the selected entry after the completion of the current waveform repetitions. • IMMediate: the sequencer jumps as soon as possible to the selected entry, without waiting for the completion of the current waveform repetitions.
Returns	AFTERrepetitions | IMMediate
Example	AWGControl:JUMPMode IMMediate Sets the Jump Mode parameter to immediate jump. AWGControl:JUMPMode? Might return IMMEediate

Table 44: AWGControl:JUMPMode

Command	AWGControl:DJStrobe (No Query Form)
Description	This command sends the pattern strobe event to the sequencer.
Group	Control
Syntax	AWGControl:DJStrobe <value>
Related Commands	SEQuence:ELEM[n]:PATTERN SEQuence:ELEM[n]:PATTERNJUMPTOMode SEQuence:ELEM[n]:PATTERNJUMPTOEntry
Arguments	<value> := <NR1> between 0 and 255
Returns	None
Example	AWGControl:DJStrobe 123 Sends 123 as pattern strobe event.

Table 45: AWGControl:DJStrobe

2.11 Calibration And Diagnostic Commands

Command	CALibration[:ALL]
Description	This command does a full calibration of the AWG. In its query form, the command does a full calibration and returns a status indicating the success or failure of the operation. This command is equivalent to the *CAL? command.
Group	Control
Syntax	CALibration[:ALL] CALibration[:ALL]?
Related Commands	*CAL?
Arguments	
Returns	<calibration error code> ::= <NR1> 0 indicates no error –1 indicates an error
Example	CALIBRATION:ALL performs a calibration. CALIBRATION:ALL? performs a calibration and returns results. For example, it might return 0, indicating that the calibration completed without any errors.

Table 46: CALibration[:ALL]

Command	DIAGnostic[:ALL]
Description	This command executes the self diagnostic procedure. The query form of this command executes the self diagnostic procedure and returns the results in the form of numeric of values of 0 for no errors or -1 for one or more tests failed.
Group	Control
Syntax	DIAGnostic[:ALL] DIAGnostic[:ALL]?
Related Commands	*CAL?
Arguments	
Returns	A single <NR1> value. 0 indicates no error. –1 indicates that the test failed.

Example	DIAGNOSTIC executes the self diagnostic procedure. DIAGNOSTIC? executes the self diagnostic procedure and might return 0, indicating that there are no errors.

Table 47: DIAGnostic[:ALL]

2.12 Output Group Commands

Command	OUTPut[n]:BLOFfset
Description	This command and query sets or returns the Base Line Offset parameter of the analog channel "n".
Group	Output
Syntax	OUTPut[n]:BLOFfset {MINimum | MAXimum | DEFault | <Volts>} OUTPut[n]:BLOFfset? [{MINimum | MAXimum}]
Related Commands	None
Arguments	• MINimum: sets the parameter to the minimum value • MAXimum: sets the parameter to the maximum value • DEFault: sets the parameter to the default value • <Volts> := <NRf> value The value of n indicates the channel number.
Returns	<NRf>
Example	OUTPut1:BLOFfset MAXimum Sets the channel 1 Base Line Offset to the maximum value. OUTPut1:BLOFfset ? Might return 6V

Table 48: OUTPut[n]:BLOFfset

Command	OUTPut[n]:DELay
Description	This command and query sets or returns the Skew parameter of the analog channel "n".
Group	Output
Syntax	OUTPut[n]:DELay {MINimum | MAXimum | DEFault | <Seconds>} OUTPut[n]:DELay? [{MINimum | MAXimum}]
Related Commands	None
Arguments	• MINimum: sets the parameter to the minimum value • MAXimum: sets the parameter to the maximum value

	• DEFault: sets the parameter to the default value • < Seconds > := <NRf> value The value of n indicates the channel number.
Returns	<NRf>
Example	OUTPut1:DELay 100 n Sets the channel 1 skew to OUTPut1:DELay ? Might return 99.999E-9

Table 49: OUTPut[n]:DELay

Command	OUTPut[n]:POLarity
Description	This command and query sets or returns the Polarity of the analog channel "n".
Group	Output
Syntax	OUTPut[n]:POLarity {NORMal | INVerted} OUTPut[n]:POLarity?
Related Commands	None
Arguments	• NORMal: sets the polarity to positive • INVerted: inverts the output polarity The value of n indicates the channel number.
Returns	NORM | INV
Example	OUTPut1:POLarity Inverts the channel 1 polarity OUTPut1: POLarity? Might return INV that means the polarity is inverted.

Table 50: OUTPut[n]:POLarity

Command	OUTPut[n]:SCALe
Description	This command and query sets or returns the Amplitude Scale parameter of the analog channel "n". This parameter can be modified at run-time to adjust the waveform amplitude while the instrument is running and it is applied to all the waveforms contained in the sequencer. It is expressed in % and it has a range of 0% to 100%. 100% means that the waveform keeps its original amplitude.
Group	Output
Syntax	OUTPut[n]:SCALe {MINimum | MAXimum | DEFault | <Percentage>}

	OUTPut[n]:SCALe? [{MINimum | MAXimum}]
Related Commands	None
Arguments	• MINimum: sets the parameter to the minimum value • MAXimum: sets the parameter to the maximum value • DEFault: sets the parameter to the default value • < Percentage > := <NRf> value The value of n indicates the channel number.
Returns	<NRf>
Example	OUTPut1: SCALe 70 Sets the amplitude scale for the channel 1 to 70% OUTPut1: SCALe? Might return 70

Table 51: OUTPut[n]:SCALe

Command	OUTPut[n]:SERIESIMPedance
Description	This command and query sets or returns the Output Impedance of the analog channel "n".
Group	Output
Syntax	OUTPut[n]:SERIESIMPedance {50Ohm | LOW} OUTPut[n]:SERIESIMPedance?
Related Commands	None
Arguments	• 50Ohm: the output impedance is set to 50 Ohm • LOW: the output impedance is set to Low impedance (5 Ohm) The value of n indicates the channel number.
Returns	<NRf>
Example	OUTPut1:SERIESIMPedance 50Ohm Sets the output impedance for the channel 1 to 50 Ohm OUTPut1:SERIESIMPedance? Might return LOW that means the channel 1 output impedance is set to Low value.

Table 52: OUTPut[n]:SERIESIMPedance

Command	OUTPut[n][:STATe]
Description	This command and query sets or returns the Output state of the analog channel "n". Setting the output state of a channel to ON will switch on its analog output signal.

Group	Output
Syntax	OUTPut[n][:STATe] {OFF | ON} OUTPut[n][:STATe]?
Related Commands	None
Arguments	• OFF: the selected output is disconnected from the DUT by means of a relay and the analog signal is turned off. • ON: the selected output is turned on The value of n indicates the channel number.
Returns	0 | 1 where '0' means OFF and '1' means ON
Example	OUTPut1 ON Turns on the analog output signal and closes the output relay

Table 53: OUTPut[n][:STATe]

Command	DIGitals:LEVel[m]
Description	This command and query sets or returns the Voltage Level of the digital output probe.
Group	Output
Syntax	DIGitals:LEVel[m] {MINimum | MAXimum | DEFault | <Volts>} DIGitals:LEVel[m]? [{MINimum | MAXimum}]
Related Commands	None
Arguments	• MINimum: sets the parameter to the minimum value • MAXimum: sets the parameter to the maximum value • DEFault: sets the parameter to the default value • < Volts > := <NRf> value The value of m indicates the digital probe where m=1 Digital Pod A, m=2 Digital Pod B, m=3 Digital Pod C, m=4 Digital Pod D
Returns	<NRf>
Example	DIGitals:LEVel2 2.5 Set the voltage amplitude to 2.5V for the pod B DIGitals:LEVel2? Might return 2.5

Table 54: DIGitals:LEVel[m]

Command	DIGitals:NUMber
Description	This command and query sets or returns the available number of the digital channels. The maximum number of available digital lines depends on the AWG model and on the installed license.

	There are up to 8 digital lines every 2 analog channels.
Group	Output
Syntax	DIGitals:NUMber{0 | 2 | 4 | 6 | 8 | 10 | 12 | 14 | 16 | 18 | 20 | 22 | 24 | 26 | 28 | 30 | 32} DIGitals:NUMber? Note: enabling the digital lines will cause a decrease of resolution in the analog output channels as shown in the user manual table (Digital Channels section).
Related Commands	None
Arguments	<value>::=<NR1> The <value> indicates the available number of the digital channels.
Returns	<value>
Example	DIGitals:NUMber 2 DIGitals: NUMber? Might returns "2"

Table 55: DIGitals:NUMber

Command	DIGitals:SKEW[m]
Description	This command and query sets or returns the Skew of the digital output probe. It sets the delay between the analog channels and the digital channels in order to de-skew the analog and digital outputs. The skew between analog/digital channels depends on the sampling frequency: the minimum skew is 1 clock cycle @ the sampling frequency.
Group	Output
Syntax	DIGitals:SKEW[m] {MINimum | MAXimum | DEFault | <Seconds>} DIGitals:SKEW[m]? [{MINimum | MAXimum}]
Related Commands	None
Arguments	• MINimum: sets the parameter to the minimum value • MAXimum: sets the parameter to the maximum value • DEFault: sets the parameter to the default value • < Seconds > := <NRf> value The value of m indicates the digital probe where m=1 Digital Pod A, m=2 Digital Pod B, m=3 Digital Pod C, m=4 Digital Pod D
Returns	<NRf>

Example	DIGitals:SKEW1 100n Set the skew for the digital probe A to 100n DIGitals:SKEW1? Might return 100.020E-9 (The set value is automatically rounded to the closest value that the hardware can implement).

Table 56: DIGitals:SKEW[m]

Command	DIGitals:STATe
Description	This command and query sets or returns the state of the digital outputs. It enables or disables the available digital channels. **Note**: if the Digital Channels number is set to '0', DIGitals:STATe ON has no effect.
Group	Output
Syntax	DIGitals:STATe {OFF | ON} DIGitals:STATe?
Related Commands	None
Arguments	OFF | ON • OFF: disables the digital lines • ON: enables the digital lines
Returns	0 | 1 where '0' means OFF and '1' means ON
Example	DIGitals:STATe ON Enables all the available digital channels. DIGitals:STATe? Might returns "1" that means ON

Table 57: DIGitals:STATe

2.13 Display Group Commands

Command	DISPlay:FOCus (No Query Form)
Description	Selects the channel that is displayed "in front" on a two/four/eight-channel instrument.
Group	Display
Syntax	DISPlay:FOCus {CH1 | CH2 | CH3 | CH4 | CH5 | CH6 | CH7 | CH8 | DIGitals}
Related Commands	None

Arguments	• CH1 | CH2 | CH3 | CH4 | CH5 | CH6 | CH7 | CH8 is the display page relative to the selected analog channel • DIGitals: digital channels display page
Returns	None
Example	DISPlay:FOCus CH2 It brings the channel 2 page to the front

Table 58: DISPlay:FOCus

Command	DISPlay:UNIT:VOLT (No Query Form)
Description	Selects the method for specifying voltage ranges. You can specify a voltage range as an amplitude and an offset or as high and low values.
Group	Display
Syntax	DISPlay:UNIT:VOLT {AMPLitudeoff | HIGHlow}
Related Commands	SEQuence:ELEM[n]:AMPlitude[m] SEQuence:ELEM[n]:OFFset[m] SEQuence:ELEM[n]:VOLTage:HIGH[m] SEQuence:ELEM[n]:VOLTage:LOW[m]
Arguments	None
Returns	None
Example	The following examples both specify a waveform voltage from 1 to 4 V: DISP:UNIT:VOLT HIGH; SEQ:ELEM:VOLT:LOW 1; SEQ:ELEM:VOLT:HIGH 4; DISP:UNIT:VOLT AMPL; SEQ:ELEM:AMP 3; SEQ:ELEM:OFF 2.5;

Table 59: DISPlay:UNIT:VOLT

Command	DISPlay[:WINDow]:TEXT:CLEar (No Query Form)
Description	This command clears the text message from the display screen.
Group	Display
Syntax	DISPlay[:WINDow]:TEXT:CLEar
Related Commands	DISPlay[:WINDow]:TEXT[:DATA]
Arguments	
Returns	None
Example	The following example writes "Hello" on the display and clears it.

	DISPlay:TEXT "Hello" DISPlay:TEXT:CLEar It clears Hello

Table 60: DISPlay[:WINDow]:TEXT:CLEar

Command	DISPlay[:WINDow]:TEXT[:DATA] The DISPlay[:WINDow]:TEXT[:DATA] command displays a text message on the instrument screen. The DISPlay[:WINDow]:TEXT[:DATA]? query returns the text string currently displayed on the instrument screen.
Description	This command clears the text message from the display screen.
Group	Display
Syntax	DISPlay[:WINDow]:TEXT[:DATA] <message> DISPlay[:WINDow]:TEXT[:DATA]?
Related Commands	DISPlay[:WINDow]:TEXT:CLEar
Arguments	<message> ::= <string>
Returns	None
Example	The following example writes “Hello” on the display and clears it. DISPlay:TEXT "Hello" DISPlay:TEXT:CLEar It clears Hello

Table 61: DISPlay[:WINDow]:TEXT[:DATA]

Command	HCOPy:SDUMp[:IMMediate] (No Query Form)
Description	This command copies the current screen shot image and saves it to a specified file in the file system.
Group	HCopy
Syntax	HCOPy:SDUMp[:IMMediate] <file_path>
Related Commands	MMEMory:CDIRectory, MMEMory:MSIS
Arguments	<file_path> must be a valid path with file name and extension. Valid extensions are: "bmp", "jpg", "jpeg, "gif", "png", "tiff". It can be absolute or relative path, if it is a relative path it will be combined with the specified path with the commands MMEMory:CDIRectory and MMEMory:MSIS.
Returns	None
Example	HCOPY:SDUMP[:IMMEDIATE] "D:\my_screenshot.png"

	Copies the screen shot image and creates a graphic file on the USB memory (D:) called screenshot.png.

Table 62: HCOPy:SDUMp[:IMMediate]

2.14 License Group Commands

Command	LICense:ERRor (Query Only)
Description	This query-only command returns a code about license options loading operation.
Group	License
Syntax	LICense:ERRor?
Related Commands	
Arguments	None
Returns	A single <NR1> value. 0 indicates no error. –1 indicates a fail condition.
Example	LICense:ERRor? Might return '0' that means no errors has occurred during the options loading procedure.

Table 63: LICense:ERRor

Command	LICense:HID (Query Only)
Description	This command returns the instrument HostID unique identifier.
Group	License
Syntax	LICense:HID?
Related Commands	
Arguments	None
Returns	<string> The instrument HostID unique identifier.
Example	LICense:HID? Might return "T0302I18450099"

Table 64: LICense:HID

Command	LICense: INSTall (No Query Form)

Description	This command accepts a license and installs it on the instrument. Restarting the instrument may be necessary to fully activate the additional capabilities.
Group	License
Syntax	LICense:INSTall <license_string>
Related Commands	
Arguments	<license_string> ::= <string>
Returns	None
Example	LICense:INSTall <license_string> Install license file to unit.

Table 65: LICense: INSTall

Command	LICense:LIST (Query Only)
Description	This query-only command returns the license codes as a comma-separated list of string. If no license is installed an error occurs.
Group	License
Syntax	LICense:LIST?
Related Commands	
Arguments	None
Returns	<license_code>[,<license_code> [,<license_code>]]] <license_code> ::= <string>
Example	LICense:LIST? Might return "02f0-4fff-b528-6ef7-f501-4515-8f38-1f54"

Table 66: LICense:LIST

Command	*OPT (Query Only)
Description	This command returns the installed options and application licenses for the AWG.
Group	License
Syntax	*OPT?
Related Commands	
Arguments	
Returns	<opt> ::= <option_description>: <option_value> <option_description> ::= <string> <option_value> ::= <string>

Example	*OPT? might return the following string: "Memory Option: 128 MSample, Digitals Option: No Digitals, Amplitude Option: 6 VPP"

Table 67: *OPT

2.15 Marker Group Commands

Command	MARKer:LEVel[m]
Description	This command and query sets or returns the marker output Voltage Level parameter in Volts. The marker voltage is calculated on 50 ohm load; the minimum value is 1V@50 Ohm load and the maximum value is 2.5V@50 ohm load.
Group	Marker
Syntax	MARKer:LEVel[m] {MINimum | MAXimum | DEFault | <Volts>} MARKer:LEVel[m]? [{MINimum | MAXimum}]
Related Commands	
Arguments	• MINimum: sets the parameter to the minimum value • MAXimum: sets the parameter to the maximum value • DEFault: sets the parameter to the default value • < Volts > := <NRf> value The value of m indicates the Marker Out number: there is one marker every 2 analog channels.
Returns	<NRf>
Example	MARKer:LEVel1 2.3 Sets the marker 1 level to 2.3V MARKer:LEVel1? MAX Might return 2.5

Table 68: MARKer:LEVel[m]

Command	MARKer:MODE[m]
Description	This command and query sets or returns the marker output "m" Marker Mode parameters.

Group	Marker
Syntax	MARKer:MODE[m] {FIXEDLow | FIXEDHigh | AUTOmatic | REPLYdigital} MARKer:MODE[m]?
Related Commands	
Arguments	• FIXEDLow: the marker level is fixed to low level • FIXEDHigh: the marker level is fixed to high level • AUTOmatic: the marker has a behavior that depends on the Run Mode (please check *Marker Settings* section on TrueArb user Manual for a more in depth explanation). • REPLYdigital: it means that The Marker Out will behave like the Digital line 0 output. This choice is available only when the digital option is installed and the Digital Channels number is >0 The value of m indicates the Marker Out number: there is one marker every 2 analog channels.
Returns	FIXEDLow | FIXEDHigh | AUTOmatic | REPLYdigital
Example	MARKer:MODE1 AUTOmatic Sets the marker 1 mode to automatic. If the instrument is in Continuous, it generates a Marker pulse of the duration from 6ns to 12 ns (depending on the sampling frequency), synchronous with the analog outputs, for each sequencer entry and for each repetition. MARKer:MODE1? Might return FIXEDHigh

Table 69: MARKer:MODE[m]

Command	MARKer:SKEW[m]
Description	This command and query sets or returns the marker delay. It defines the skew between the marker and the analog channels. The edited value is automatically rounded to the closest value that the hardware can implement.
Group	Marker
Syntax	MARKer:SKEW[m] {MINimum | MAXimum | DEFault | <Seconds>} MARKer:SKEW[m]? [{MINimum | MAXimum}] **Important Note:** The maximum value you can set in the marker skew depends on the Sampling rate.
Related Commands	None

Arguments	• MINimum: sets the parameter to the minimum value • MAXimum: sets the parameter to the maximum value • DEFault: sets the parameter to the default value • < Volts > := <NRf> value The value of m indicates the Marker Out number.
Returns	<NRf>
Example	MARKer:SKEW1 1E-9 Sets the marker delay to 1ns MARKer:SKEW1? Might returns "1E-3"

Table 70: MARKer:SKEW[m]

2.16 Clock Group Commands

Command	ROSCillator
Description	This command and query sets or returns the frequency of the Reference Clock parameter.
Group	Clock
Syntax	ROSCillator {MINimum | MAXimum | DEFault | <Herts>} ROSCillator? [{MINimum | MAXimum}]
Related Commands	None
Arguments	<Hertz> ::= <NRf> A single <NRf> value.
Returns	A single <NR3> value.
Example	ROSCillator 10E6 Set a frequency of the Reference Clock to10 MHz.

Table 71: ROSCillator

Command	AWGControl:SRATe
Description	This command sets or returns the sample rate for the Sampling Clock.
Group	Clock
Syntax	AWGControl:SRATe {MINimum | MAXimum | DEFault | <Hertz>} AWGControl:SRATe? [{MINimum | MAXimum}]
Related Commands	ROSCillator ROSCillator:SOURce
Arguments	<Hertz> ::= <NRf>

	A single <NRf> value. *RST sets this to the maximum value.
Returns	A single <NR3> value.
Example	AWGControl:SRATe 5E8 sets the clock sample rate to 500 MHz. CLOCK:SRATE? might return 1.20000000000E+9, indicating the clock sample rate is set to 1.2 GHz.

Table 72: AWGControl:SRATe

Command	ROSCillator:SOURce
Description	This command sets the reference clock to either internal or external.
Group	Clock
Syntax	ROSCillator:SOURce {INTernal | EXTernal} ROSCillator:SOURce?
Related Commands	ROSCillator ROSCillator:SOURce
Arguments	• INTernal: means that the reference clock is set to Internal • EXTernal: means that the reference clock is set to External
Returns	INT | EXT
Example	ROSCillator:SOURce INTernal Selects the internal clock reference ROSCillator:SOURce? Might return INT.

Table 73: ROSCillator:SOURce

2.17 IEEE Mandated and Optional Group Commands

Command	*CAL (Query Only)
Description	This query runs the self calibration procedure and returns a status code indicanting the success or failure of self calibration. During the self calibration the AWG panel will be locked. *CAL? is equivalent to the CALibration[:ALL] command
Group	IEEE Mandated and Optional Group Command
Syntax	*CAL?
Related Commands	CALibration[:ALL]
Arguments	
Returns	A single <NR1> value, {0 | -1}

Example	*CAL? might return -1 on any failure, 0 on all pass.

Table 74: *CAL

Command	*CLS (No Query Form)
Description	This command sets or returns the status of Event Status Enable Register (ESER). (See Status and events chapter.)
Group	IEEE Mandated and Optional Group Command
Syntax	*CLS
Related Commands	None
Arguments	
Returns	
Example	*CLS Clears all the event registers and queues.

Table 75: *CLS

Command	*ESE
Description	This command sets or returns the status of Event Status Enable Register (ESER). (See Status and events chapter.)
Group	IEEE Mandated and Optional Group Command
Syntax	*ESE <NR1> *ESE?
Related Commands	*CLS, *ESR?, *SRE, *STB?
Arguments	A single <NR1> value.
Returns	A single <NR1> value.
Example	*ESE 177 sets the ESER to 177 (binary 10110001), which sets the PON, CME, EXE, and OPC bits. *ESE? might return 177.

Table 76: *ESE

Command	*ESR (Query Only)
Description	This command returns the status of Standard Event Status Register (SESR). (See Status and events chapter.)
Group	IEEE Mandated and Optional Group Command
Syntax	*ESR?
Related Commands	*CLS *ESE *SRE

	*STB
Arguments	
Returns	A single <NR1> value.
Example	*ESR? might return 181, which indicates that the SESR contains the binary number 10110101.

Table 77: *ESR

Command	*IDN (Query Only)
Description	This command returns identification information for the AWG. Refer to Std IEEE 488.2 for additional information.
Group	IEEE Mandated and Optional Group Command
Syntax	*IDN
Related Commands	
Arguments	
Returns	<Manufacturer>, <Model>, <Serial number>, <Software version> <Manufacturer>:: = ACTIVE TECHNOLOGIES <Model>:: = XXXXXXX (indicates the actual instrument model number) <Serial number>:: = XXXXXXX (indicates the actual serial number) <Firmware version>:: = SCPI:99.0,SV:x.x.x.x (x.x.x.x is software version)
Example	*IDN? returns <Manufacturer>, <Model>, <Serial number>, <Software version> Might return ACTIVE TECHNOLOGIES,AWG4012,T03020I18450099,SCPI:1999.0,SV:1.0.34.0

Table 78: *IDN

Command	*OPC
Description	This command causes the AWG to sense the internal flag referred to as the "No-Operation-Pending" flag. The command sets bit 0 in the Standard Event Status Register when pending operations are complete. The query form returns a "1" when the last overlapping command operation is finished.
Group	IEEE Mandated and Optional Group Command
Syntax	*OPC *OPC?
Related Commands	*WAI
Arguments	

Returns	A single <NR1> value.
Example	*OPC? returns 1 to indicate that the last issued overlapping command is finished.

Table 79: *OPC

Command	*RST (No Query Form)
Description	This command resets the AWG to its default state. The AWG will be stopped.
Group	IEEE Mandated and Optional Group Command
Syntax	*RST
Related Commands	
Arguments	
Returns	
Example	*RST resets the AWG.

Table 80: *RST

Command	*SRE
Description	This command sets or returns the bits in the Service Request Enable register. (See Status and events chapter.)
Group	IEEE Mandated and Optional Group Command
Syntax	*SRE <enable_value> *SRE?
Related Commands	*CLS *ESE *ESR *STB
Arguments	A single <NR1> value.
Returns	A single <NR1> value.
Example	*SRE 48 sets the bits in the SRER to the binary value 00110000. *SRE? might return a value of 32, showing that the bits in the SRER have the binary value 00100000.

Table 81: *SRE

Command	*TRG (No Query Form)
Description	This command generates a trigger event. This is equivalent to press and release the trigger button on the front panel.

Group	IEEE Mandated and Optional Group Command
Syntax	*TRG
Related Commands	TRIGger[:IMMediate], AWGControl:RMODe
Arguments	
Returns	
Example	*TRG generates a trigger event.

Table 82: *TRG

Command	*TST (Query Only)
Description	This command executes the Self Diagnostic procedure.
Group	IEEE Mandated and Optional Group Command
Syntax	*TST?
Related Commands	DIAGnostic[:IMMediate], DIAGnostic:DATA?, DIAGnostic:RESult?
Arguments	
Returns	A single <NR1> value. A returned value of 0 indicates no error.
Example	*TST? might return –1, indicating that the Self Diagnostic procedure failed.

Table 83: *TST

Command	*WAI (No Query Form)
Description	This command is used to ensure that the previous command is complete before the next command is issued.
Group	IEEE Mandated and Optional Group Command
Syntax	*WAI
Related Commands	*OPC
Arguments	
Returns	None
Example	

Table 84: *WAI

2.18 Memory Group Commands

Command	*RCL (No Query Form)

Description	This command restores the state of the instrument from a copy of the settings stored in the setup memory. The settings are stored using the *SAV command. If the specified setup memory is deleted, this command causes an error.
Group	Memory
Syntax	*RCL {0|1|2|3|4}
Related Commands	*SAV
Arguments	0, 1, 2, 3, or 4 Specifies the location of setup memory
Returns	None
Example	*RCL 3 Restores the instrument from a copy of the settings stored in memory location 3

Table 85: *RCL

Command	*SAV (No Query Form)
Description	This command stores the current AWG configuration to a specified setup memory location. The current AWG configuration is automatically saved every time the user changes a parameter. This configuration, here named "current configuration" is automatically loaded at the power on. Note: If a specified numbered setup memory is locked, this command causes an error.
Group	Memory
Syntax	*SAV {0|1|2|3|4}
Related Commands	*RCL
Arguments	0, 1, 2, 3, or 4 Specifies the location of setup memory
Returns	None
Example	*SAV 2 Saves the current instrument state in the memory location 2

Table 86: *SAV

Command	MEMory:NSTates (Query Only)
Description	This command returns the total number of available configurations saved in the AWG.

Group	Memory
Syntax	MEMory:NSTates?
Related Commands	
Arguments	
Returns	A single <NR1> value
Example	MEMory:NSTates? Might return 2

Table 87: MEMory:NSTates

Command	MEMory:STATe:CATalog (Query Only)
Description	List the names of available configurations saved in the AWG.
Group	Memory
Syntax	MEMory:STATe:CATalog?
Related Commands	
Arguments	
Returns	<cfg_name>[,<cfg_name>[,<cfg_name>[,<cfg_name>]]] <cfg_name> ::= <string>
Example	MEMory:STATe:CATalog? Configuration_20190110_183606_568,test

Table 88: MEMory:STATe:CATalog

Command	MEMory:STATe:DELete (No Query Form)
Description	This command deletes the specified configuration. If the specified configuration is locked or not present this command causes an error.
Group	Memory
Syntax	MEMory:STATe:DELete {0|1|2|3|4|<cfg_name>}
Related Commands	
Arguments	0, 1, 2, 3, 4 or <cfg_name> <cfg_name> ::= <string> Specifies the configuration. Configuration 0 refers to "MEM_0" configuration, configuration 1 refers to "MEM_1" configuration, configuration 2 refers to "MEM_2" configuration, configuration 3 refers to "MEM_3" configuration and configuration 4 refers to "MEM_4" configuration.
Returns	
Example	MEMory:STATe:DELete 1 Deletes the configuration "MEM_1".

Table 89: MEMory:STATe:DELete

Command	MEMory:STATe:LOCK
Description	This command sets or queries whether to lock the specified configuration. If you lock a configuration, you cannot overwrite or delete the setup file.
Group	Memory
Syntax	MEMory:STATe:LOCK {0|1|2|3|4|<cfg_name>},{OFF|ON} MEMory:STATe:LOCK? {0|1|2|3|4|<cfg_name>}
Related Commands	
Arguments	0, 1, 2, 3, 4 or <cfg_name>, where <cfg_name> ::= <string>, specifies the configuration to locked or queried. ON or <NR1>≠0 Locks the specified configuration. OFF or <NR1>=0 Allows you to overwrite or delete the specified configuration.
Returns	A single <NR1>
Example	MEMORY:STATE:LOCK 1,ON Locks the configuration "MEM_1"

Table 90: MEMory:STATe:LOCK

Command	MEMory:STATe:NAME
Description	This command copies a configuration with "src_name" in "dst_name" configuration. If a configuration with "dst_name" is already present an error will occur. The query command returns the names refferred to the configurations 0, 1, 2, 3, 4. The names will be ever the same: "MEM_0", "MEM_1", "MEM_2", "MEM_3", "MEM_4".
Group	Memory
Syntax	MEMory:STATe:NAME {0|1|2|3|4|<src_cfg_name>},{0|1|2|3|4|<dst_cfg_name>} MEMory:STATe:NAME? {0|1|2|3|4}
Related Commands	
Arguments	0, 1, 2, 3, 4 or <src_cfg_name>, where <src_cfg_name> ::= <string>, specifies the source configuration name. 0, 1, 2, 3, 4 or <dst_cfg_name>, where <dst_cfg_name> ::= <string>, specifies the destination configuration name.
Returns	<cfg_name> where <cfg_name> ::= <string>

Example	MEM:STAT:NAME 1,"TEST_RACK_1" Copy location "MEM_1" in "TEST_RACK_1"

Table 91: MEMory:STATe:NAME

Command	MEMory:STATe:VALid (Query Only)
Description	This command returns the availability of a configuration.
Group	Memory
Syntax	MEMory:STATe:VALid? {0|1|2|3|4|<src_cfg_name>}
Related Commands	
Arguments	
Returns	<NR1> 1 means that the specified configuration is present. 0 means that the specified configuration is not present.
Example	MEMORY:STATE:VALID? 0 might return 1 if the specified setup memory has been saved.

Table 92: MEMory:STATe:VALid

Command	DELete:SETUp (No Query Form)
Description	This command deletes a configuration.
Group	Memory
Syntax	DELete:SETUp <cfg_name>
Related Commands	
Arguments	<cfg_name>::=<string>
Returns	
Example	DELete:SETUp "test" Deletes the configuration named test.

Table 93: DELete:SETUp

Command	RECALL:SETUp (No Query Form)
Description	This command recalls a configuration. This command is equivalent to AWGControl:SREStore.
Group	Memory
Syntax	RECALL:SETUp <cfg_name>
Related Commands	
Arguments	<cfg_name>::=<string>
Returns	

Example	RECALL:SETUp “test” Restores the instrument configuration named “test”.

Table 94: RECALL:SETUp

2.19 Mass Memory Group Commands

Command	MMEMory:CATalog (Query only)
Description	This command returns a list of informations concerning the contents of the current directory of the file system on the AWG. In particular it returns the list of all files and directories present there specifying their names and sizes. It reports also the dimension of free space of the mass storage in bytes.
Group	Mass memory
Syntax	MMEMory:CATalog[:ALL]?
Related Commands	MMEMory:CDIRectory MMEMory:MSIS
Arguments	None
Returns	<NR1>,<file_entry>,<file_entry>,<file_entry>,….. where: <NR1> indicates the free space of the mass storage in bytes, <file_entry> ::= "<file_name>,<file_type>,<file_size>" where: <file_name> ::= <string> is the exact name of the file, <file_type> ::= is DIR for an entry that is a directory, empty/blank otherwise, <file_size> ::= <NR1> is the size of the file in bytes. For <file_type> marked DIR, the file size will always be 0.
Example	MMEMory:CATalog? It might return: 3878652,"SAMPLE1.ZIP,,2948","aaa.txt,,1024","MY_WAVES,DIR,0"

Table: MMEMory:CATalog

Command	MMEMory:CDIRectory
Description	This command sets or returns the current directory of the file system on the AWG. This command is strictly related to MMEMory:MSIS command that permits to set the mass storage unit <msus> namely the unit used by all MMEMory commands.

	The current directory for the programmatic interface is different from the currently selected directory in the Windows Explorer on the AWG. **NOTE:** Only removable units and "C:\Users\<username>\Pictures\Saved_Pictures" directory are accessible by MMEMory commands.
Group	Mass memory
Syntax	MMEMory:CDIRectory [<directory_name>] MMEMory:CDIRectory?
Related Commands	None
Arguments	<directory_name> ::= <string>
Returns	<directory_name>
Example	Assuming the current <msus> is "C:" and current directory "C:\Users\<username>\Pictures\Saved Pictures". MMEMORY:CDIRECTORY "configurations" changes the current directory to "C:\Users\<username>\Pictures\Saved Pictures\configurations". MMEMORY:CDIRECTORY "tmp" changes the current directory to "D:\tmp" MMEMORY:CDIRECTORY? returns "tmp" if the current directory is "D:\tmp". MMEMORY:CDIRECTORY changes the current directory to "\" if <msus> is a removable device, or "Users\<username>\Pictures\Saved Pictures" if <msus> is "C:"

Table 95: MMEMory:CDIRectory

Command	MMEMory:COPY (No Query Form)
Description	This command copies source_ file into target_file. The file names must include any file extension. If target_file already exists it will be overwritten. **NOTE 1:** Only removable units and "C:\Users\<username>\Pictures\Saved_Pictures" directory are accessible by MMEMory commands.

	NOTE 2: Only files with the following extension can be copied: .jpeg, .png, .bmp, .jpg, .gif, .tiff, .zip, .txt, .trc, .bin.
Group	Mass memory
Syntax	MMEMory:COPY <source_file>,<target_file>
Related Commands	None
Arguments	<source_file>::= <string> <target_file>::= <string> Both of them could be absolute or relative (respect current directory) paths.
Returns	None
Example	Assuming the current <msus> is "C:" and current directory "C:\Users\<username>\Pictures\Saved Pictures". MMEMORY:COPY "source.txt","D:\My_Waves\target.txt" copies "source.txt" located in "C:\Users\<username>\Pictures\Saved Pictures" directory in "target.txt" located in "D:\My_Waves" directory .

Table 96: MMEMory:COPY

Command	MMEMory:DATA
Description	This command sets or returns block data to/from a file in the current mass storage device. The file path may contain a full file path. However, if the file path only contains a file name, the current directory is assumed. **NOTE 1:** Only removable units and "C:\Users\<username>\Pictures\Saved_Pictures" directory are accessible by MMEMory commands. **NOTE 2:** Only files with the following extension can be manipulated: .jpeg, .png, .bmp, .jpg, .gif, .tiff, .zip, .txt, .trc, .bin.
Group	Mass memory
Syntax	MMEMory:DATA <file_name>,<start_index>,<block_data> MMEMory:DATA? <file_name>[,<start_index>[,<size>]]
Related Commands	MMEMory:CDIRectory, MMEMory:MSIS
Arguments	<file_name> ::= <string> could be absolute or relative path.

	<start_index> ::= <NR1> is the index of byte of the desired <file_name> where writing/reading operations will start. <size> ::= <NR1> is the size, in bytes, to write/read. <block_data> ::= see Block Data Format chapter (IEEE 488.2 data block).
Returns	<block_data> ::= IEEE 488.2 block data format.
Example	Assuming the current <msus> is "C:" and current directory "C:\Users\<username>\Pictures\Saved Pictures". MMEMORY:DATA "123.TXT",#13ABC loads "ABC" into 123.TXT in the current directory. MMEMory:DATA "data.txt",1024,#2048XXXXXX… inserts 2048 bytes specified by XXXXXX… values of in file "data.txt" in the current directory. MMEMory:DATA? "D:\tmp\waveform.txt",2048,1024 returns #41024XXXX…. where XXXX…. are 1024 bytes of file "waveform.txt" located in "D:\tmp" starting from 2048-th byte.

Table 97: MMEMory:DATA

Command	MMEMory:DATA:SIZE (Query Only)
Description	This command returns the size in bytes of a selected file. **NOTE 1:** Only removable units and "C:\Users\<username>\Pictures\Saved_Pictures" directory are accessible by MMEMory commands. **NOTE 2:** Only files with the following extension can be manipulated: .jpeg, .png, .bmp, .jpg, .gif, .tiff, .zip, .txt, .trc, .bin.
Group	Mass memory
Syntax	MMEMory:DATA:SIZE? <file_name>
Related Commands	MMEMory:CDIRectory, MMEMory:MSIS
Arguments	<file_name> ::= <string> it could be absolute or relative path.
Returns	<NR1> is the size, in bytes, of the selected file
Example	Assuming the current <msus> is "C:" and current directory "C:\Users\<username>\Pictures\Saved Pictures".

	MMEMORY:DATA:SIZE? "data.bin" might return 1024. MMEMORY:DATA:SIZE? "D:\tmp\waveform.txt" might return 65535.

Table 98: MMEMory:DATA:SIZE

Command	MMEMory:DELete (No Query Form)
Description	This command deletes a file or directory from the AWG's accessible files system. **NOTE 1:** Only removable units and "C:\Users\<username>\Pictures\Saved_Pictures" directory are accessible by MMEMory commands. **NOTE 2:** Only files with the following extension can be deleted: .jpeg, .png, .bmp, .jpg, .gif, .tiff, .zip, .txt, .trc, .bin.
Group	Mass memory
Syntax	MMEMory:DELete <file_name>[,<msus>]
Related Commands	MMEMory:CDIRectory, MMEMory:MSIS
Arguments	<file_name> ::= <string> could be absolute or relative path. <msus> (mass storage unit specifier) ::= <string>
Returns	None
Example	Assuming the current <msus> is "C:" and current directory "C:\Users\<username>\Pictures\Saved Pictures". MMEMORY:DELETE "data.bin" deletes “data.bin” from the current directory. MMEMORY:DELETE "\my\proj\awg\test.txt","D:" deletes “test.txt” from “D:\my\proj\awg” directory.

Table 99: MMEMory:DELete

Command	MMEMory:DOWNload:DATA (No Query Form)
Description	Downloads data from the host computer to a file in the instrument. The filename must have been previously specified by MMEMory:DOWNload:FNAMe. The data in <binary_block> is written to the select file, and any data previously stored in the file is lost.

	NOTE: Only removable units and "C:\Users\<username>\Pictures\Saved Pictures" directory are accessible by MMEMory commands.
Group	Mass Memory
Syntax	MMEMory:DOWNload:DATA <binary_block>
Related Commands	MMEMory:DOWNload:FNAMe
Arguments	Any IEEE-488 definite or indefinite block
Returns	None
Example	Writes the word "Hello" to the file "D:\Myfile.txt" on internal storage. MMEM:DOWN:FNAM "D:\Myfile.txt" MMEM:DOWN:DATA #15Hello

Table 100: MMEMory:DOWNload:DATA

Command	MMEMory:DOWNload:FNAMe (No Query Form)
Description	Creates or opens the specified filename prior to writing data to that file with MMEMory:DOWNload:DATA. **NOTE**: Only removable units and "C:\Users\<username>\Pictures\Saved Pictures" directory are accessible by MMEMory commands.
Group	Mass Memory
Syntax	MMEMory:DOWNload:FNAMe <filename>
Related Commands	MMEMory:DOWNload:DATA
Arguments	Any valid file name
Returns	None
Example	Writes the word "Hello" to the file "D:\Myfile.txt" on the internal flash file system. MMEM:DOWN:FNAM "D:\Myfile.txt" MMEM:DOWN:DATA #15Hello

Table 101: MMEMory:DOWNload:FNAMe

Command	MMEMory:EXPort (No Query Form)
Description	This command exports a waveform from the current waveform list to an archive file (.zip). **NOTE 1 .**zip is a zip file with Active Technologies format.

	NOTE 2: It's not possible export a predefined waveform from waveform list. **NOTE 3:** If the archive file name is already present in the destination directory, the file will be overwritten. **NOTE4:** Only removable units and "C:\Users\<username>\Pictures\Saved_Pictures" directory are accessible by MMEMory commands. The archive path may contain a full file path. However, if the file path only contains an archive name, the archive waveform will be exported starting from the current directory. **NOTE 5:** This operation is the same as that it can be done through the software menu: Wave. List -> Export.
Group	Mass memory
Syntax	MMEMory:EXPort <wfm_name>,<archive_name>
Related Commands	None.
Arguments	<wfm_name>::=<string> is a name in waveform list, <archive_name>::=<string>.zip could be absolute or relative path. The extension .zip must always be specified in archive name.
Returns	None
Example	Assuming the current <msus> is "C:" and current directory "C:\Users\<username>\Pictures\Saved Pictures". MMEMORY:EXPORT "sine1024","d:\waveforms\arc.zip" exports a waveform named "sine1024" in to "d:\waveform\arc.zip" file.

Table 102: MMEMory:EXPort

Command	MMEMory:IMPort (No Query Form)
Description	This command imports a file into the current configuration waveforms list. **NOTE 1:** It's possible import a file only with the following extensions: .txt, .trc and .zip - Active Technologies format-.

	NOTE 2: If a .zip file is imported, the parameter {ANAlog | DIGitals} will be not taken into consideration. The .zip format already contains this information. **NOTE 3:** If the waveform name (wfm_name) has already present in waveform list, an error will be occurred. **NOTE 4:** If waveform type {ANAlog | DIGitals} isn't specify, ANAlog type will be assumpted as default. **NOTE 5:** Only removable units and "C:\Users\<username>\Pictures\Saved_Pictures" directory are accessible by MMEMory commands. The file path may contain a full file path. However, if the file path only contains a file name, the current directory is assumed as destination directory. **NOTE 6:** This operation is the same as that it can be done through the software menu: Wave. List -> Import.
Group	Mass memory
Syntax	MMEMory:IMPort <wfm_name>,<archive_name>[,{ANAlog | DIGitals}]
Related Commands	MMEMory:OPEN
Arguments	<wfm_name>::=<string>, <archive_name> ::=<string>.{zip | trc | txt} could be absolute or relative path.
Returns	None
Example	Assuming the current <msus> is "C:" and current directory "C:\Users\<username>\Pictures\Saved Pictures". MMEMORY:IMPORT "Test1","D:\TestFiles\waveform1.zip" imports waveform1.zip file in waveform Test1 of waveform list (its type is already specified in .zip file). MMEMORY:IMPORT "Test2"," waveform2.txt",DIG imports waveform2.txt file in waveform Test2 of waveform list (its type will be digital).

Table 103: MMEMory:IMPort

Command	MMEMory:LOAD:ALL (No Query Form)
Description	This command uploads a complete configuration present in an archive (.zip) and applay it in place of the current configuration. This operation is performed in two steps: 1. First the archive is loaded into the configurations list. Note: • The name of the archive file is used as the name of the imported configuration (my_configuration.zip => my_configuration). • If a configuration with the same name is already present, the configuration is replaced. • If a configuration with the same name is already present and is located then an error occurs. • This operation is the same as that it can be done through the software menu: More -> Load From -> Import. 2. Once the archive has been imported into the configuration list, the new configuration will be loaded as current configuration. **NOTE 1:** Only removable units and "C:\Users\<username>\Pictures\Saved_Pictures" directory are accessible by MMEMory commands. The archive path may contain a full file path. However, if the file path only contains an archive name, the archive will be searched starting from the current directory. **NOTE 2:** When this command is executed the AWG must be in idle state.
Group	Mass memory
Syntax	MMEMory:LOAD:ALL <archive_name>
Related Commands	None
Arguments	<archive_name> ::= <string>.zip specifies a configuration file to be loaded. It could be absolute or relative path.
Returns	None
Example	Assuming the current <msus> is "C:" and current directory "C:\Users\<username>\Pictures\Saved Pictures". MMEM:LOAD:ALL "configurations\my_configuration.zip" loads a complete instrument setup from the file "my_configuration.zip" located in "C:\Users\<username>\Pictures\Saved Pictures\configurations" directory.

Table 104: MMEMory:LOAD:ALL

Command	MMEMory:LOAD:STATe (No Query Form)
Description	This command imports a configuration saved in an archive (.zip) into the configurations list with the specified name. **NOTE 1:** If a configuration with the same name is already present in configuration list then it will be replaced. **NOTE 2:** If a configuration with the same name is already present in configuration list and it's locked then an error will occur. **NOTE 3:** Only removable units and "C:\Users\<username>\Pictures\Saved_Pictures" directory are accessible by MMEMory commands. The archive path may contain a full file path. However, if the file path only contains an archive name, the archive will be searched starting from the current directory. **NOTE 4:** This operation is equivalent to what the user can do through the following user interface menu: Other -> Load from -> Import
Group	Mass memory
Syntax	MMEMory:LOAD:STATe <archive_name>,{0|1|2|3|4|<cfg_name>}
Related Commands	MEMory:STATe:LOCK, MMEMory:STORe:STATe
Arguments	<archive_name>::=<string>.zip specifies a configuration file to be loaded. It could be absolute or relative path. It possible specify the configuration like: - 0|1|2|3|4: configuration 0 refers to "MEM_0" configuration, configuration 1 refers to "MEM_1" configuration... - <cfg_name>::=<string>: spcecifies the configuration name where will be saved the loaded configuration.
Returns	None
Example	Assuming the current <msus> is "C:" and current directory "C:\Users\<username>\Pictures\Saved Pictures". MMEMORY:LOAD:STATE "SETUP1.zip",1 loads the configuration file "SETUP1.zip" located in the current directory in "MEM_1".

Table 105: MMEMory:LOAD:STATe

Command	MMEMory:MDIRectory (No Query Form)
Description	This command creates a new directory in the current path on the mass storage system. **NOTE**: Only removable units and "C:\Users\<username>\Pictures\Saved Pictures" directory are accessible by MMEMory commands.
Group	Mass Memory
Syntax	MMEMory:MDIRectory <directory_name>
Related Commands	MMEMory:CDIRectory, MMEMory:MSIS
Arguments	<directory_name> ::= <string>
Returns	None
Example	MMEMory:MDIRectory "Waveform" creates the directory "Waveform" in the current directory.

Table 106: MMEMory:LOAD:STATe

Command	MMEMory:MOVE (No Query Form)
Description	Moves <file1> to <file2>. The file names must include the file extension. The file path may contain a full file path. However, if the file path only contains a file name, the file will be searched starting from the current directory. If <file2> already exists it will be overwritten. **NOTE1**: Only removable units and "C:\Users\<username>\Pictures\Saved Pictures" directory are accessible by MMEMory commands. **NOTE 2:** Only files with the following extension can be moved: .jpeg, .png, .bmp, .jpg, .gif, .tiff, .zip, .txt, .trc, .bin.
Group	Mass Memory
Syntax	MMEMory:MOVE <file1>,<file2>
Related Commands	None
Arguments	<file1>,<file2> they could be absolute or relative paths.
Returns	None
Example	MMEM:MOVE "MySetup.zip","D:\Backup.zip"

	MMEM:MOVE "D:\arbMonday.txt", "D:\arbTuesday.bin"

Table 107: MMEMory:LOAD:STATe

Command	MMEMory:MSIS
Description	This command selects or returns a mass storage device used by all MMEMory commands. <msus> specifies a drive using a drive letter. The drive letter can represent hard disk drives, USB memory. **NOTE 1**: Every time the <msus> changes the current directory is resetted to the root folder. **NOTE 2:** If the new <msus> is the "C:" unit, the current directory is resetted to "C:\Users\<username>\Pictures\Saved Pictures" folder.
Group	Mass Memory
Syntax	MMEMory:MSIS [<msus>] MMEMory:MSIS?
Related Commands	
Arguments	<msus> (mass storage unit specifier)::= <string>
Returns	<msus> NOTE. If the mass storage device has not been defined, the returned <msus> value is the system's default drive which is typically the :C drive.
Example	MMEMORY:MSIS? might return "E:", assuming the current MSUS is the E: drive. MMEMORY:MSIS "D:" changes the MSUS to the D: drive where D: is a USB memory

Table 108:MMEMory:MSIS

Command	MMEMory:OPEN (No Query Form)
Description	This command imports a waveform stored in a file (.zip, .txt, .trc) into the waveform list of the current AWG's. **NOTE 1**: If waveform file has ".zip" extension then the parameter {ANAlog | DIGitals} will not be taken into account. The ".zip" format already has this information in itself. **NOTE 2:** The waveform name will be derived directly from the ar-chive name.

	NOTE 3: If the waveform name is already present in the waveform list then an error will occur. **NOTE 4**: If the waveform type {ANAlog | DIGitals} is not specified, the ANAlog type will be assumed as default. **NOTE 5:** Only removable units and "C:\Users\<username>\Pictures\Saved_Pictures" directory are accessible by MMEMory commands. The waveform path may contain a full file path. However, if the file path only contains a waveform name, the waveform will be searched starting from the current directory. **NOTE 6:** This operation is equivalent to what the user can do through the following user interface menu: Wave. List -> Import
Group	Mass memory
Syntax	MMEMory:OPEN <filepath>[,{ANAlog | DIGitals}]
Related Commands	None
Arguments	<filepath> ::= <string>.{zip | txt | trc} could be an absolute or relative path.
Returns	None
Example	Assuming the current <msus> is "C:" and current directory "C:\Users\<username>\Pictures\Saved Pictures". MMEMORY:OPEN "D:\TestFiles\WFM_001.txt",DIG after command execution, if no errors occur, digital waveform WFM_001 will be appear into the AWG's waveform list. WFM_001 was originally located in "D:\TestFiles" directory.

Table 109: MMEMory:OPEN

Command	MMEMory:OPEN:SETup (No Query Form)
Description	This command uploads a complete configuration present in an archive (.zip) and applay it in place of the current configuration. **NOTE 1**: It's an alias of MMEMory:LOAD:ALL command. See its description for more informations. **NOTE 2**: The AWG must be in idle state.
Group	Mass memory

Syntax	MMEMory:OPEN:SETup <filepath>
Related Commands	None
Arguments	<filepath> ::= <string>.zip could be an absolute or relative path.
Returns	None
Example	MMEMORY:OPEN:SETUP "D:\TestFiles\mySetup.zip" opens and apply the setup file mySetup.zip.

Table 110: MMEMory:OPEN:SETup

Command	MMEMory:RDIRectory (No Query Form)
Description	This command removes an empty directory (folder) from the mass storage system. **NOTE**: Only removable units and "C:\Users\<username>\Pictures\Saved Pictures" directory are accessible by MMEMory commands. The directory path may contain a full path. However, if only directory name is specified, the directory will be searched starting from the current directory.
Group	Mass Memory
Syntax	MMEMory:RDIRectory <folder>
Related Commands	None
Arguments	<folder>:==<string> could be an absolute or relative path.
Returns	None
Example	If the current directory is "C:\Users\<username>\Pictures\Saved Pictures" MMEMory:RDIRectory "Test" removes the empty folder "Test" from the current directory.

Table 111: MMEMory:RDIRectory

Command	MMEMory:STORe:ALL (No Query Form)
Description	This command saves the current AWG's configuration in an archive (.zip). **NOTE 1**: If an archive with the same name and path is already present then the archive is overwritten.

	NOTE 2: Only removable units and "C:\Users\<username>\Pictures\Saved Pictures" directory are accessible by MMEMory commands. The archive path may contain a full path. However, if only name is specified, the archive will be saved starting from the current directory.
Group	Mass memory
Syntax	MMEMory:STORe:ALL <filename>
Related Commands	None
Arguments	<ffilename> ::= <string>.zip could be an absolute or relative path.
Returns	None
Example	Assuming the current <msus> is "C:" and current directory "C:\Users\<username>\Pictures\Saved Pictures". MMEMory:STORe:ALL "D:\mySetup.zip" saves the current istrument's configuration in mySetip.zip file in drive D:.

Table 112: MMEMory:STORe:ALL

Command	MMEMory:SAVE:SETup (No Query Form)
Description	This command saves the current AWG's configuration in an archive (.zip). **NOTE 1**:It's an alias of MMEMory:STORe:ALL command. See its description for more informations.
Group	Mass memory
Syntax	MMEMory:SAVE:SETup <filename>
Related Commands	None
Arguments	<ffilename> ::= <string>.zip could be an absolute or relative path.
Returns	None
Example	Assuming the current <msus> is "C:" and current directory "C:\Users\<username>\Pictures\Saved Pictures". MMEMORY:SAVE:SETUP " mySetup.zip" saves the current istrument's configuration in "C:\Users\<username>\Pictures\Saved Pictures\ mySetip.zip " file.

Table 113: MMEMory:SAVE:SETup

Command	MMEMory:STORe:STATe (No Query Form)

Description	This command saves a configuration present in the configurations list in an archive (.zip). **NOTE 1**: If an archive with the same path and name is already present then it will be overwritten. **NOTE 2**: Only removable units and "C:\Users\<username>\Pictures\Saved Pictures" directory are accessible by MMEMory commands. The archive path may contain a full path. However, if only name is specified, the archive will be saved starting from the current directory.
Group	Mass memory
Syntax	MMEMory:STORe:STATe {0|1|2|3|4|<cfg_name>}, <archive_name>
Related Commands	MMEMory:LOAD:STATe, MMEMory:LOCK[:STATe]
Arguments	It possible specify the configuration like: - 0|1|2|3|4: configuration 0 refers to "MEM_0" configuration, configuration 1 refers to "MEM_1" configuration... - <cfg_name>::=<string> <archive_name>::=<string>.zip specifies the file path. It could be an absolute or relative path.
Returns	None
Example	Assuming the current <msus> is "C:" and current directory "C:\Users\<username>\Pictures\Saved Pictures". MMEMORY:STORE:STATE 1,"setup1.zip"copies the configuraion stored in the setup memory location MEM_1 to a file named "setup1.zip" in the current directory.

Table 114: MMEMory:STORe:STATe

Command	MMEMory:UPLoad (Query Only)
Description	This command returns the contents of a file. **NOTE 1:** Only removable units and "C:\Users\<username>\Pictures\Saved_Pictures" directory are accessible by MMEMory commands.

	NOTE 2: Only files with the following extension can be deleted on removable units: .jpeg, .png, .bmp, .jpg, .gif, .tiff, .zip, .txt, .trc, .bin.
Group	Mass memory
Syntax	MMEMory:UPLoad? <file_name>
Related Commands	MMEMory:CDIRectory, MMEMory:MSIS
Arguments	<file_name> ::= <string> could be absolute or relative paths.
Returns	<block_data>::= IEEE 488.2 block data format
Example	MMEM:UPL? "D:\Myfile.zip" returns the contents of "Myfile.zip".

Table 115: MMEMory:UPLoad

2.20 Status Group Commands

Command	STATus:OPERation:CONDition (Query Only)
Description	This query returns the contents of the Operation Condition Register.
Group	Status
Syntax	STATus:OPERation:CONDition?
Related Commands	STATus:OPERation:ENABle, STATus:OPERation[:EVENt]?
Arguments	
Returns	A single <NR1> value showing the contents of the OCR.
Example	STATus:OPERation:CONDition? might return 0, showing that the bits in the OCR have the binary value 0000000000000000.

Table 116: STATus:OPERation:CONDition

Command	STATus:OPERation:ENABle
Description	This command and query sets or returns the mask for the Operation Enable Register. **NOTE**: The most-significant bit cannot be set true.
Group	Status
Syntax	STATus:OPERation:ENABle <enable_value> STATus:OPERation:ENABle?
Related Commands	STATus:OPERation:CONDition?, STATus:OPERation[:EVENt]?
Arguments	A single <NR1>::=<enable_value> value. Range: 0 to 65535
Returns	A single <NR1> value.

Example	STATus:OPERation:ENABle 1 enables the Calibrating bit. STATus:OPERation:ENABle? might return 1, showing that the bits in the OENR have the binary value 00000000 00000001, which means that the Calibrating bit is valid.

Table 117: STATus:OPERation:ENABle

Command	STATus:OPERation[:EVENt] (Query Only)
Description	This query returns the contents of Operation Event Register. Reading the OEVR clears it.
Group	Status
Syntax	STATus:OPERation[:EVENt]?
Related Commands	STATus:OPERation:CONDition?, STATus:OPERation:ENABle
Arguments	
Returns	A single <NR1> value showing the contents of the OEVR.
Example	STATus:OPERation? might return 1, showing that the bits in the OEVR have the binary value 00000000 00000001, which means that the CALibrating bit is set.

Table 118: STATus:OPERation[:EVENt]

Command	STATus:PRESet (No Query Form)
Description	This command sets the Operation Enable Register (OENR) and Questionable Enable Register (QENR).
Group	Status
Syntax	STATus:PRESet
Related Commands	None
Arguments	
Returns	
Example	STATUS:PRESET resets the SCPI enable registers.

Table 119: STATus:PRESet

Command	STATus:QUEStionable:CONDition (Query Only)
Description	This query returns the status of the Questionable Condition Register. Note that the QCR is not used in the waveform generator.
Group	Status
Syntax	STATus:QUEStionable:CONDition?
Related Commands	STATus:QUEStionable:ENABle, STATus:QUEStionable[:EVENt]?
Arguments	

Returns	A single <NR1> value.
Example	STATus:QUEStionable:CONDition? Might return 0.

Table 120: STATus:QUEStionable:CONDition

Command	STATus:QUEStionable:ENABle
Description	This command sets or returns the enable mask of the Questionable Enable Register (QENR) which allows true conditions in the Questionable Event Register to be reported in the summary bit. Refer to the Status and event reporting system section for additional information.
Group	Status
Syntax	STATus:QUEStionable:ENABle <enable_value> STATus:QUEStionable:ENABle?
Related Commands	STATus:QUEStionable:CONDition?, STATus:QUEStionable[:EVENt]?
Arguments	<enable_value>::= <NR1> is the enable mask of the QENR. Range: 0 to 65535.
Returns	A single <NR1> value showing the contents of the QENR.
Example	STATus:QUEStionable:ENABle 64 enables the FREQuency bit. STATus:QUEStionable:ENABle? might return 64, showing that the bits in the QENR have the binary value 00000000 00100000, which means that the FREQuency bit is valid.

Table 121: STATus:QUEStionable:ENABle

Command	STATus:QUEStionable[:EVENt] (Query Only)
Description	This command returns the contents of the Questionable Event Register (QEVR). Reading the QEVR clears it. Refer to the Status and event reporting system section for additional information.
Group	Status
Syntax	STATus:QUEStionable[:EVENt]?
Related Commands	STATus:QUEStionable:CONDition?, STATus:QUEStionable:ENABle
Arguments	
Returns	A single <NR1> value showing the contents of the QEVR.
Example	STATus:QUEStionable: EVENt? might return 64, showing that the bits in the QEVR have the binary value 00000000 00100000, which means that the FREQuency bit is set.

Table 122: STATus:QUEStionable[:EVENt]

Command	*STB (Query Only)
Description	This query returns the contents of Status Byte Register.
Group	Status
Syntax	*STB?
Related Commands	*CLS, *ESE, *ESR?, *SRE
Arguments	
Returns	A single <NR1> value.
Example	*STB? Might return 96, which indicates that the SBR contains the binary number 0110 0000.

Table 123: *STB

Command	*PSC
Description	This command sets and queries the power-on status flag that controls the automatic power-on execution of SRER and ESER. When *PSC is true, SRER and ESER are set to 0 at power-on. When *PSC is false, the current values in the SRER and ESER are preserved in nonvolatile memory when power is shut off and are restored at power-on.
Group	Status
Syntax	*PSC {0|1} *PSC?
Related Commands	
Arguments	<NR1>=0 Sets the power-on status clear flag to false, disables the power-on clear, and allows the instrument to possibly assert SRQ after power-on. <NR1>≠0 Sets the power-on status clear flag true; sending *PSC 1 therefore enables the power-on status clear and prevents any SRQ assertion after power-on
Returns	A single <NR1> value.
Example	PSC 0 Sets the power-on status clear flag to false.

Table 124: *PSC

2.21 System Group Commands

Command	SYSTem:BEEPer:STATe
Description	The SYSTem:BEEPer:STATe command sets the beeper ON or OFF. The SYSTem:BEEPer:STATe? command returns "0" (OFF) or "1" (ON). When the beeper is set to ON, the instrument will beep when an error message or a warning message is displayed on the screen. The instrument does not beep when an error or warning caused by remote command execution.
Group	System
Syntax	SYSTem:BEEPer:STATe {OFF | ON} SYSTem:BEEPer:STATe?
Related Commands	
Arguments	ON or <NR1>≠0 Enables the beeper OFF or <NR1>=0 Disables the beeper
Returns	A single <NR1> value.
Example	SYSTem:BEEPer:STATe ON Enables the beeper function

Table 125: SYSTem:BEEPer:STATe

Command	SYSTem:BEEPer[:IMMediate]
Description	This command causes the instrument to beep immediately.
Group	System
Syntax	SYSTem:BEEPer[:IMMediate]
Related Commands	
Arguments	ON or <NR1>≠0 Enables the beeper OFF or <NR1>=0 Disables the beeper
Returns	
Example	SYSTem:BEEPer Generates an audible beep.

Table 126: SYSTem:BEEPer[:IMMediate]

Command	SYSTem:DATE

Description	This command sets or returns the system date.
Group	System
Syntax	SYSTem:DATE <yyyy>,<mm>,<dd> SYSTem:DATE?
Related Commands	
Arguments	<year>::=<NR1> (Four digit number) <month>::=<NR1> from 1 to 12 <day>::=<NR1> from 1 to 31
Returns	<year>,<month>,<day>
Example	SYSTem:DATE 2012,11,20 sets the date to November 20, 2012.

Table 127: SYSTem:DATE

Command	SYSTem:ERRor[:NEXT] (Query Only)
Description	This command returns data from the error and event queues.
Group	System
Syntax	SYSTem:ERRor[:NEXT]?
Related Commands	
Arguments	<Error number>, <error description> Error number ::=<NR1>. error description ::=<string>.
Returns	<year>,<month>,<day>
Example	SYSTem:ERRor:NEXT? might return 0,"No error", indicating there are not errors.

Table 128: SYSTem:ERRor[:NEXT]

Command	SYSTem:KLOCk[:STATe]
Description	This command locks or unlocks the instrument front panel controls. The query command returns "0" (OFF) or "1" (ON).
Group	System
Syntax	SYSTem:KLOCk[:STATe] {OFF | ON} SYSTem:KLOCk[:STATe]?
Related Commands	
Arguments	ON or <NR1>≠0 Locks front panel controls OFF or <NR1>=0 Unlocks front panel controls
Returns	<NR1>
Example	SYSTem:KLOCk ON

	Locks front panel controls.

Table 129: SYSTem:KLOCk[:STATe]

Command	SYSTem:SECurity:IMMediate (No Query Form)
Description	This command erases all configurations and user waveforms and recalls the factory default settings. Calibration data is not erased.
Group	System
Syntax	SYSTem:SECurity:IMMediate
Related Commands	
Arguments	
Returns	<NR1>
Example	SYSTem:SECurity:IMMediate Initializes the instrument.

Table 130: SYSTem:SECurity:IMMediate

Command	SYSTem:TIME
Description	This command sets or returns the system time (hours, minutes and seconds).
Group	System
Syntax	SYSTem:TIME <hh>,<mm>,<ss> SYSTem:TIME?
Related Commands	
Arguments	<hour>,<minute>,<second> <hour> ::= <NR1> specifies the hours. Range: 0 to 23. <minute> ::= <NR1> specifies the minutes. Range: 0 to 59. <second> ::= <NR1> specifies the seconds. Range: 0 to 59.
Returns	<hour>,<minute>,<second> <hour> ::= <NR1> specifies the hours. <minute> ::= <NR1> specifies the minutes. <second> ::= <NR1> specifies the seconds.
Example	SYSTem:TIME 10,15,30 sets the time to 10:15:30. SYSTem:TIME? might return 12,20,32, indicating the system time is 12:20:32.

Table 131: SYSTem:TIME

Command	SYSTem:VERSion (Query Only)

Description	This command returns the SCPI version number to which the command conforms.
Group	System
Syntax	SYSTem:VERSion?
Related Commands	
Arguments	
Returns	A single <NR2> value. <NR2> ::= YYYY.V where YYYY is the year version and V is revision number for that year.
Example	SYSTem:VERSion? Might return 1999.0

Table 132: SYSTem:VERSion

2.22 Trigger Group Commands

Command	ABORt (No Query Form)
Description	This command stops waveform playout when the Run Mode is set to Gated. This is equivalent to release the Force Trigger button on the front panel when the instrument is in gated mode
Group	Trigger
Syntax	ABORt
Related Commands	AWGControl:RMODe, *TRG
Example	ABORt It will stop the waveform playout on all channels with their Run Mode set to Gated.

Table 133: ABORt

Command	TRIGger[:SEQuence]:SOURce
Description	This command sets or returns the instrument trigger source.
Group	Trigger
Syntax	TRIGger[:SEQuence]:SOURce {TIMer | EXTernal | MANual} TRIGger[:SEQuence]:SOURce?
Related Commands	None
Arguments	<source>::= {TIMer | EXTernal | MANual} TIMer: the trigger is sent at regular intervals. EXTernal: the trigger come from the external BNC connector.

	MANual: the trigger is sent via software or using the trigger button on front panel.
Returns	TIM | EXT | MAN
Example	TRIGger:SOURce TIMer It sets the trigger source to timer. TRIGger:SOURce? Might return TIMer

Table 134: TRIGger[:SEQuence]:SOURce

Command	TRIGger[:SEQuence]:SLOPe
Description	This command sets or returns the instrument trigger input slope for the external source.
Group	Trigger
Syntax	TRIGger[:SEQuence]:SLOPe {POSitive | NEGative | BOTH} TRIGger[:SEQuence]:SLOPe?
Related Commands	TRIGger[:SEQuence]:SOURce
Arguments	<slope>::= {POSitive | NEGative | BOTH} POSitive specifies a trigger event on the rising edge of the external trigger signal. NEGative specifies a trigger event on the falling edge of the external trigger signal. BOTH specifies that a trigger event occurs both on falling and rising edge of the external trigger signal.
Returns	<slope>
Example	TRIGger:SLOPe POS It sets the trigger slope to rising edge. TRIGger: SLOPe? Might return BOTH.

Table 135: TRIGger[:SEQuence]:SLOPe

Command	TRIGger[:SEQuence]:LEVel
Description	This command sets or queries the threshold of the trigger-in signal.
Group	Trigger
Syntax	TRIGger[:SEQuence]:LEVel {MINimum | MAXimum | DEFault | <Volts>} TRIGger[:SEQuence]:LEVel? [{MINimum | MAXimum}]
Related Commands	TRIGger[:SEQuence]:SOURce
Arguments	<Volts>::= <NRf> specifies the voltage threshold of external trigger-in signal.

	MINimum sets or queries the minimum threshold level. MAXimum sets or queries the maximum threshold level. DEFault sets the default threshold level (0V).
Returns	<threshold>
Example	TRIGger:LEVel 5.5 Sets the external trigger-in threshold level to 5.5 V. TRIGger:LEVel? MINimum It returns -10.

Table 136: TRIGger[:SEQuence]:LEVel

Command	TRIGger[:SEQuence]:TIMer
Description	This command sets or queries the interval of an internal trigger-in event when you select TIMER as the trigger source with the TRIGger[:SEQuence]:SOURce command.
Group	Trigger
Syntax	TRIGger[:SEQuence]:TIMer {MINimum | MAXimum | DEFault | <Seconds>} TRIGger[:SEQuence]:TIMer? [{MINimum | MAXimum}]
Related Commands	TRIGger[:SEQuence]:SOURce
Arguments	<seconds>::=<NRf>[<units>] where: <units>::=[µs | ms | s] and it specifies the interval value expressed in <units>. MINimum sets or queries the minimum interval value. MAXimum sets or queries the maximum interval value. DEFault sets the default interval value (1 s).
Returns	<Seconds>
Example	TRIGger:TIMer 300ms It sets the trigger-in interval to 0.3 seconds. TRIGger:TIMer? It might return 5 seconds.

Table 137: TRIGger[:SEQuence]:TIMer

Command	TRIGger:IMPedance
Description	This command sets or returns the trigger input impedance. It applies only to the external trigger signal (from BNC connector).
Group	Trigger
Syntax	TRIGger:IMPedance {50Ohm | 1KOhm}] TRIGger:IMPedance?

Related Commands	TRIGger[:SEQuence]:SOURce
Arguments	<impedance>::= {50Ohm | 1KOhm}
Returns	< impedance >
Example	TRIGger: IMPedance 50Ohm It sets the trigger impedance to 50 Ohm TRIGger: IMPedance? It might return 50 Ohm

Table 138: TRIGger:IMPedance

Command	TRIGger[:SEQuence][:IMMediate] (No Query Form)
Description	This command forces a trigger event to occur.
Group	Trigger
Syntax	TRIGger[:SEQuence][:IMMediate]
Related Commands	AWGControl:RMODe ABORt *TRG
Arguments	None
Returns	None
Example	TRIGger:SEQuence:IMMediate It generates a trigger event.

Table 139: TRIGger[:SEQuence][:IMMediate]

2.23 Sequence Group Commands

Command	SEQuence:ELEM[n]:AMPlitude[m]
Description	Sets or returns the voltage peak-to-peak amplitude for the element "n" of the channel "m"
Group	Sequence
Syntax	SEQuence:ELEM[n]:AMPlitude[m] {MINimum | MAXimum | DEFault | <Volts>} SEQuence:ELEM[n]:AMPlitude[m]? [{MINimum | MAXimum}]
Related Commands	SEQuence:ELEM[n]:OFFset[m]
Arguments	The value of n indicates the sequence element number. The value of m is the channel number. <Volts>::=<NRf> sets the peak-to peak amplitude level expressed in Volt.

	MINimum sets or queries the minimum amplitude level. MAXimum sets or queries the maximum amplitude level. DEFault sets the default amplitude level (2V).
Returns	<Volts>::=<NRf>
Example	SEQuence:ELEM3:AMPlitude2 4 Sets the voltage amplitude of the channel 2 for sequencer element 3 to 4. SEQuence:ELEM3:AMPlitude2? Might returns 4.

Table 140: SEQuence:ELEM[n]:AMPlitude[m]

Command	SEQuence:ELEM[n]:OFFset[m]
Description	Sets or returns the voltage offset for the element "n" of the channel "m"
Group	Sequence
Syntax	SEQuence:ELEM[n]:OFFset[m] {MINimum | MAXimum | DEFault | <Volts>} SEQuence:ELEM[n]:OFFset[m]? [{MINimum | MAXimum}]
Related Commands	SEQuence:ELEM[n]:AMPlitude[m]
Arguments	The value of n indicates the sequence element number. The value of m is the channel number. <Volts>::=<NRf> sets the offset level expressed in Volt. MINimum sets or queries the minimum offset level. MAXimum sets or queries the maximum offset level. DEFault sets the default offset level (0V).
Returns	<Volts>::=<NRf>
Example	SEQuence:ELEM5:OFFset2 1.2 Sets the offset of the channel 2 for sequencer element 5 to 1.2 V SEQuence:ELEM5:OFFset2? MAXimum Might returns 2.5.

Table 141: SEQuence:ELEM[n]:OFFset[m]

Command	SEQuence:ELEM[n]:VOLTage:HIGH[m]
Description	This command sets or returns the high voltage level of the waveform for the element "n" of the channel "m".
Group	Sequence
Syntax	SEQuence:ELEM[n]:VOLTage:HIGH[m] {MINimum | MAXimum | DEFault | <Volts>}

	SEQuence:ELEM[n]:VOLTage:HIGH[m]? [{MINimum | MAXimum}]
Related Commands	SEQuence:ELEM[n]:VOLTage:LOW[m]
Arguments	The value of n indicates the sequence element number. The value of m is the channel number. <Volts>::=<NRf> sets the is the high level of output amplitude expressed in Volt. MINimum sets or queries the minimum value of high voltage level. MAXimum sets or queries the maximum value of high voltage level. DEFault sets the default value of high voltage level (1V).
Returns	<Volts>::=<NRf>
Example	SEQuence:ELEM2:VOLTage:HIGH1 2.3 Sets the high level voltage of the channel 1 for sequencer element 2 to 2.3 V SEQuence:ELEM2:VOLTage:HIGH1? Might returns 2.3.

Table 142: SEQuence:ELEM[n]:VOLTage:HIGH[m]

Command	SEQuence:ELEM[n]:VOLTage:LOW[m]
Description	This command sets or returns the low voltage level of the waveform for the element "n" of the channel "m".
Group	Sequence
Syntax	SEQuence:ELEM[n]:VOLTage:LOW[m] {MINimum | MAXimum | DEFault | <Volts>} SEQuence:ELEM[n]:VOLTage:LOW[m]? [{MINimum | MAXimum}]
Related Commands	SEQuence:ELEM[n]:VOLTage:HIGH[m]
Arguments	The value of n indicates the sequence element number. The value of m is the channel number. <Volts>::=<NRf> sets the is the low level of output amplitude expressed in Volt. MINimum sets or queries the minimum value of low voltage level. MAXimum sets or queries the maximum value of low voltage level. DEFault sets the default value of low voltage level (-1V).
Returns	<Volts>::=<NRf>
Example	SEQuence:ELEM2:VOLTage:LOW2 -1.7 Sets the low level voltage of the channel 2 for sequencer element 2 to -1.7 V SEQuence:ELEM1:VOLTage:LOW3? MINimum Might returns -6.

Table 143: SEQuence:ELEM[n]:VOLTage:LOW[m]

Command	SEQuence:ELEM[n]:LENGth
Description	This command sets or returns the number of samples of the waveform for the element "n".
Group	Sequence
Syntax	SEQuence:ELEM[n]:LENGth {MINimum | MAXimum | DEFault | <value>} SEQuence:ELEM[n]:LENGth? [{MINimum | MAXimum}]
Related Commands	None
Arguments	The value of n indicates the sequence element number. <value>::=<NR1> is the length of entry n. MINimum sets or queries the minimum value of length. MAXimum sets or queries the maximum value of length. DEFault sets the default value of length (2048).
Returns	<value>::=<NR1>
Example	SEQuence:ELEM3:LENGth 100 Sets the sequencer length of the element n to 100 for all channels SEQuence:ELEM2:LENGth? Might returns 1000.

Table 144: SEQuence:LENgth

Command	SEQuence:ELEM[n]:LOOP:COUNt
Description	This command sets or returns the number of repetitions of the waveform for the element "n".
Group	Sequence
Syntax	SEQuence:ELEM[n]:LOOP:COUNt {MINimum | MAXimum | DEFault | INFinite | <value>} SEQuence:ELEM[n]:LOOP:COUNt? [{MINimum | MAXimum}]
Related Commands	None
Arguments	The value of n indicates the sequence element number. <value>::=<NR1> is the number of repetitions of entry n. MINimum sets or queries the minimum value of repetitons parameter. MAXimum sets or queries the maximum value of repetitons parameter. INFinite sets infinite repetitons. DEFault sets the default value of repetition (1).
Returns	<value>::=<NR1>
Example	SEQUENCE:ELEM1:LOOP:COUNT 120

	Sets the repetitons parameter of the element 1 to 120 for all channels SEQUENCE:ELEM1:LOOP:COUNT? MAXimum It returns 4294967295.

Table 145: SEQuence:ELEM[n]:LOOP:COUNt

Command	SEQuence:ELEM[n]:WAVeform[m]
Description	This command sets or returns the waveform for the sequence element "n". The value of "m" indicates the channel that will output the waveform when the sequence is run. It's possible select a waveform only from those in the waveform list. In waveform list are already present 10 predefined waveform: Sine, Ramp, Square, Sync, DC, Gaussian, Lorentz, Haversine, Exp_Rise and Exp_Decay but user can import in the list others customized waveforms.
Group	Sequence
Syntax	SEQuence:ELEM[n]:WAVeform[m] <wfm_name> SEQuence:ELEM[n]:WAVeform[m]?
Related Commands	None
Arguments	The value of n indicates the element index. The value of m is the channel number. <wfm_name>::=<String> is the waveform's name.
Returns	<wfm_name>::=<String>
Example	SEQUENCE:ELEM1:WAVEFORM1 "TRIANGLE1000" Sets the "TRIANGLE1000" waveform into the first element of the sequence of channel 1. SEQuence:ELEM3:WAVeform1? Might return SQUARE.

Table 146: SEQuence:ELEM[n]:WAVeform[m]

Command	SEQuence:LENGth
Description	This command sets or returns the entire sequence length. Also note that passing a value less than the sequence's current length will cause some elements to be deleted at the end of the sequence. For example if SEQuence:LENGth? returns 100 and you subsequently send SEQuence:LENGth 21, all sequence elements except the first 20 will be deleted.
Group	Sequence
Syntax	SEQuence:LENGth {MINimum | MAXimum | DEFault | <value>} SEQuence:LENGth?

Related Commands	None
Arguments	<value>::=<NR1> it is the sequence length. The MINimum and DEFault parameters will set a length = 1
Returns	<value>::=<NR1>
Example	SEQuence:LENGth 50 Sets the sequencer length to 50 entries for all channels initializing all sequence parameters to default values. SEQuence:LENGth? Might returns 50.

Table 147: SEQuence:LENGth

Command	SEQuence:NEW (No Query Form)
Description	This command creates a new sequence with only one element with default value.
Group	Sequence
Syntax	SEQuence:NEW
Related Commands	None
Arguments	None
Returns	None
Example	SEQuence:NEW It creates a new sequence (with only one element)

Table 148: SEQuence:NEW

Command	SEQuence:FOCus (No Query Form)
Description	This command visualizes a specific element of the sequencer on the display.
Group	Sequence
Syntax	SEQuence:FOCus {MINimum | MAXimum | DEFault | <value>}
Related Commands	None
Arguments	<value>::=<NR1> indicates the element to visualize. MINimum and DEFault set the first element of the sequencer to be displayed. MAXimum sets the last element of the sequencer to be displayed.
Returns	None
Example	SEQuence:FOCus 5 The fifth element of the sequencer will be displayed.

Table 149: SEQuence:FOCus

Command	SEQuence:ELEM[n]:WAITEvent
Description	This command sets or returns the wait event type for the selected element 'n'.
Group	Sequence
Syntax	SEQuence:ELEM[n]:WAITEvent {NONE | MANual | TIMer | EXTernal} SEQuence:ELEM[n]:WAITEvent?
Related Commands	None
Arguments	<wait_event>::= {NONE | MANual | TIMer | EXTernal} NONE: the wait event is disabled. MANual: the wait event is sent via software or by pressing the front panel button. TIMer: the wait event is generated by an internal timer. EXTernal: the wait event comes from the external BNC connector
Returns	<wait_event>::= {NONE | MANual | TIMer | EXTernal}
Example	SEQuence:ELEM1:WAITEvent TIMER It sets the wait event for first element of the sequence to timer SEQuence:ELEM1:WAITEvent? Might return TIMer.

Table 150: SEQuence:ELEM[n]:WAITEvent

Command	SEQuence:ELEM[n]:GOTOMode
Description	This command sets or returns the "GOTO" command type for the selected element 'n' of the sequencer. After generating the waveform(s) specified in a sequence element, the sequencer jumps to the entry specified as the GOTO Item. This is an unconditional jump.
Group	Sequence
Syntax	SEQuence:ELEM[n]:GOTOMode {FIRST | PREVious | NEXT | LAST | ITEM} SEQuence:ELEM[n]:GOTOMode?
Related Commands	SEQuence:ELEM[n]:GOTOEntry
Arguments	<mode>::= { FIRST | PREVious | NEXT | LAST | ITEM } FIRST: the sequencer will go to the first element of the sequence PREVious: the sequencer will go to the previous element of the sequence. If the selected element is the first, it will jump to the last.

	NEXT: the sequencer will go to the next element of the sequence. If the selected element is the last, it will jump to the first. ITEM: the sequencer will go to the selected item of the sequencer defined in the command SEQuence:ELEM[n]:GOTOEntry.
Returns	<mode>::= { FIRST | PREVious | NEXT | LAST | ITEM }
Example	SEQuence:ELEM4:GOTOMode FIRST The sequencer will jump to the first element after executing the fourth element. SEQuence:ELEM4:GOTOMode? Might return FIRST

Table 151: SEQuence:ELEM[n]:GOTOMode

Command	SEQuence:ELEM[n]:GOTOEntry
Description	This command sets or returns the target entry for the "GOTO" command for the selected element 'n' of the sequencer. After generating the waveform(s) specified in a sequence element, the sequencer jumps to the entry specified as the GOTO target. This is an unconditional jump.
Group	Sequence
Syntax	SEQuence:ELEM[n]:GOTOEntry {MINimum | MAXimum | DEFault | <value>} SEQuence:ELEM[n]:GOTOEntry? [{MINimum | MAXimum}]
Related Commands	SEQuence:ELEM[n]:GOTOMode
Arguments	• MINimum: sets the parameter to the minimum value • MAXimum: sets the parameter to the maximum value • DEFault: sets the parameter to the default value • < value >::= <NR1> sets the target entry index
Returns	< value > := <NR1>
Example	SEQuence:ELEM4:GOTOMode ITEM SEQuence:ELEM4:GOTOEntry 2 The sequencer will jump to the second entry of the sequencer after executing the fourth element. SEQuence:ELEM4:GOTOEntry? Might return 2.

Table 152: SEQuence:ELEM[n]:GOTOEntry

Command	SEQuence:ELEM[n]:JUMPTOMode
Description	This command sets or returns the "Jump To" command type for the selected element 'n' of the sequencer. The "Jump To" command is a conditional jump.
Group	Sequence
Syntax	SEQuence:ELEM[n]:JUMPTOMode {FIRST | PREVious | NEXT | LAST | ITEM} SEQuence:ELEM[n]:JUMPTOMode?
Related Commands	SEQuence:ELEM[n]:JUMPTOEntry SEQuence:ELEM[n]:JUMPEvent AWGControl:JUMPMode
Arguments	<mode>::= { FIRST | PREVious | NEXT | LAST | ITEM } FIRST: the sequencer will jump to the first element of the sequence PREVious: the sequencer will jump to the previous element of the sequence. If the selected element is the first, it will jump to the last. NEXT: the sequencer will jump to the next element of the sequence. If the selected element is the last, it will jump to the first. ITEM: the sequencer will jump to the selected item of the sequencer defined in the command SEQuence:ELEM[n]:JUMPTOEntry.
Returns	<mode>::= { FIRST | PREVious | NEXT | LAST | ITEM }
Example	SEQuence:ELEM4:JUMPTOMode LAST The sequencer will jump to the last element after the jump event occurs. SEQuence:ELEM4:JUMPTOMode? Might return LAST

Table 153: SEQuence:ELEM[n]:JUMPTOMode

Command	SEQuence:ELEM[n]:JUMPEvent
Description	This command sets or returns the jump event type for the selected element 'n'.
Group	Sequence
Syntax	SEQuence:ELEM[n]:JUMPEvent {NONE | MANual | TIMer | EXTernal} SEQuence:ELEM[n]:JUMPEvent?
Related Commands	SEQuence:ELEM[n]:JUMPTOEntry SEQuence:ELEM[n]:JUMPTOMode

	AWGControl:JUMPMode
Arguments	<jump_event>::= {NONE | MANual | TIMer | EXTernal} NONE: the jump event is disabled. MANual: the jump event is sent via software or by pressing the front panel button. TIMer: the jump event is generated by an internal timer. EXTernal: the jump event comes from the external BNC connector.
Returns	< jump_event >::= {NONE | MANual | TIMer | EXTernal}
Example	SEQuence:ELEM1: JUMPEvent MANual It sets the jump event for first element of the sequence to manual SEQuence:ELEM1:JUMPEvent? Might returns MANual.

Table 154: SEQuence:ELEM[n]:JUMPEvent

Command	SEQuence:ELEM[n]:JUMPTOEntry
Description	This command sets or returns the target entry index for the "Jump To" command for the selected element 'n' of the sequencer. After generating the waveform(s) specified in a sequence element, the sequencer jumps to the entry specified as the "Jump To" target index after the event occurs. This is a conditional jump. Note: this will take effect only when SEQuence:ELEM[n]:JUMPTOMode is set to ITEM
Group	Sequence
Syntax	SEQuence:ELEM[n]:JUMPTOEntry {MINimum | MAXimum | DEFault | <value>} SEQuence:ELEM[n]:JUMPTOEntry? [{MINimum | MAXimum}]
Related Commands	SEQuence:ELEM[n]:JUMPEvent SEQuence:ELEM[n]:JUMPTOMode AWGControl:JUMPMode
Arguments	• MINimum: sets the parameter to the minimum value • MAXimum: sets the parameter to the maximum value • DEFault: sets the parameter to the default value • < value >:= <NR1> sets the target entry index

Returns	< value > := <NR1>
Example	SEQuence:ELEM4:JUMPTOMode ITEM SEQuence:ELEM4:JUMPTOEntry 6 Sets the jump target entry index to the 6th element. SEQuence:ELEM4:JUMPTOEntry? Might return 6.

Table 155: SEQuence:ELEM[n]:JUMPTOEntry

Command	SEQuence:ELEM[n]:PATTERN
Description	This command sets or returns the pattern code value for the "Pattern Jump" command for the selected element 'n' of the sequencer. **Note**: The "Pattern Jump" is a conditional jump that occurs when a Pattern code is received by the sequencer. This command sets the pattern code value, while the pattern is sent by using the command AWGControl:DJStrobe
Group	Sequence
Syntax	SEQuence:ELEM[n]:PATTERN {MINimum | MAXimum | DEFault | <value>} SEQuence:ELEM[n]:PATTERN? [{MINimum | MAXimum}]
Related Commands	SEQuence:ELEM[n]:PATTERNJUMPTOMode SEQuence:ELEM[n]:PATTERNJUMPTOEntry AWGControl:DJStrobe
Arguments	• MINimum: sets the parameter to the minimum value • MAXimum: sets the parameter to the maximum value • DEFault: sets the parameter to the default value • < value > := <NR1> sets the pattern code value
Returns	<value>::=<NR1>
Example	SEQuence:ELEM1:PATTERN 123 It sets the pattern code to 123 for first element of the sequence SEQuence:ELEM1:PATTERN? Might returns 123

Table 156: SEQuence:ELEM[n]:PATTERN

Command	SEQuence:ELEM[n]:PATTERNJUMPTOMode
Description	This command sets or returns the "Pattern Jump" command type for the selected element 'n' of the sequencer. The "Pattern Jump" command is a conditional jump.
Group	Sequence
Syntax	SEQuence:ELEM[n]:PATTERNJUMPTOMode {FIRST | PREVious | NEXT | LAST | ITEM} SEQuence:ELEM[n]:PATTERNJUMPTOMode?
Related Commands	SEQuence:ELEM[n]:PATTERN SEQuence:ELEM[n]:PATTERNJUMPTOEntry AWGControl:DJStrobe
Arguments	<mode>::= { FIRST | PREVious | NEXT | LAST | ITEM } FIRST: the sequencer will jump to the first element of the sequence PREVious: the sequencer will jump to the previous element of the sequence. If the selected element is the first, it will jump to the last. NEXT: the sequencer will jump to the next element of the sequence. If the selected element is the last, it will jump to the first. ITEM: the sequencer will jump to the selected item of the sequencer defined in the command SEQuence:ELEM[n]: PATTERNJUMPTOEntry.
Returns	<mode>::= { FIRST | PREVious | NEXT | LAST | ITEM }
Example	SEQuence:ELEM4:PATTERNJUMPTOMode LAST The sequencer will jump to the last element after the pattern jump event occurs. SEQuence:ELEM4:PATTERNJUMPTOMode? Might return LAST.

Table 157: SEQuence:ELEM[n]:PATTERNJUMPTOMode

Command	SEQuence:ELEM[n]:PATTERNJUMPTOEntry
Description	This command sets or returns the target entry index for the "Pattern Jump To" command for the selected element 'n' of the sequencer. The "Pattern Jump" is a conditional jump that occurs when a Pattern code is received by the sequencer. As soon as the sequencer receives this pattern event, it will jump to the entry selected in this command.

	Note: this will take effect only when SEQuence:ELEM[n]: PATTERNJUMPTOMode is set to ITEM.
Group	Sequence
Syntax	SEQuence:ELEM[n]:JUMPTOEntry {MINimum | MAXimum | DEFault | <value>} SEQuence:ELEM[n]:JUMPTOEntry? [{MINimum | MAXimum}]
Related Commands	SEQuence:ELEM[n]:PATTERN SEQuence:ELEM[n]:PATTERNJUMPTOEntry AWGControl:DJStrobe
Arguments	• MINimum: sets the parameter to the minimum value • MAXimum: sets the parameter to the maximum value • DEFault: sets the parameter to the default value • < value >:= <NR1> sets the target entry index
Returns	< value > := <NR1>
Example	SEQuence:ELEM4:PATTERN 123 Sets the pattern code value SEQuence:ELEM4:PATTERNJUMPTOMode ITEM Sets the pattern jump command type to Item SEQuence:ELEM4:PATTERNJUMPTOEntry 8 Sets the pattern jump target entry index to the 8th element. AWGControl:DJStrobe 123 Sends the pattern strobe event. SEQuence:ELEM4:JUMPTOEntry? Might return 6.

Table 158: SEQuence:ELEM[n]:PATTERNJUMPTOEntry

2.24 Waveform Group Commands

Command	WLISt:LIST (Query Only)
Description	Returns the name of the all waveforms in the waveform list.
Group	Waveform
Syntax	WLISt:LIST?
Related Commands	None
Arguments	None

Returns	[<waveform_name>][,<waveform_name>]… <waveform_name>::=<string>
Example	WLISt:LIST? Might returns "Sine, Ramp, Square, Sync, DC, Gaussian, Lorentz, Haversine, Exp_Rise, Exp_Decay, Zero, One, Counter, Clock, Waveform_Imported1, Waveform_imported2"

Table 159: WLISt:LIST

Command	WLISt:NAME (Query Only)
Description	This command returns the waveform name from the waveform list at the position specified by the index value.
Group	Waveform
Syntax	WLISt:NAME? <Index>
Related Commands	None
Arguments	<Index>::=<NR1>
Returns	<string>::=<wfm_name> is the waveform name specified by <index>.
Example	WLIST:NAME? 21 Might return "waveform21".

Table 160: WLISt:NAME

Command	WLISt:SIZE (Query Only)
Description	This query returns the size (number of waveforms) of the waveform list. Names of both predefined and user-imported waveforms are stored in a single list.
Group	Waveform
Syntax	WLISt:SIZE?
Related Commands	None
Arguments	None
Returns	<NR1> At *RST, this returns the number of predefined waveforms.
Example	WLIST:SIZE? Might return 14.

Table 161: WLISt:SIZE

Command	WLISt:WAVeform:DATA (Query Only)
Description	This command returns all sample of an analog or digital waveform (or a portion of them specifying StartIndex and Size parameters). In

this way it possible transfer a waveform from the waveform list to the external control program.

The returned bytes represent the waveform's sample. For analog waveforms every samples consists of 2 bytes while for digital ones of 4 bytes.
In particular for analog waveform:

Analog waveform sample – 16 bits format (DAC's sample)	
byte offset 1	byte offset 0

In particular for digital waveform:

Digital Waveform Sample			
byte offset 3 **Digital Pod D**	byte offset 2 **Digital Pod C**	byte offset 1 **Digital Pod B**	byte offset 0 **Digital Pod A**

Digital Pod A

A(7)	A(6)	A(5)	A(4)	A(3)	A(2)	A(1)	A(0)/M(0)

Digital Pod B

B(7)	B(6)	B(5)	B(4)	B(3)	B(2)	B(1)	B(0)/M(1)

Digital Pod C

C(7)	C(6)	C(5)	C(4)	C(3)	C(2)	C(1)	C(0)/M(2)

Digital Pod D

D(7)	D(6)	D(5)	D4)	D(3)	D(2)	D(1)	D(0)/M(3)

The order of bytes uses the little-endian format.

It is suggested that the user make use of the StartIndex and Size to append data in multiple queries.
Using StartIndex and Size, part of a waveform can be transferred at a time. Very large waveforms can be transferred in chunks.
Transferring large waveforms in chunks allows external programs to cancel the operation before it is completed.

Group	Waveform
Syntax	WLISt:WAVeform:DATA? <wfm_name>[,<StartIndex>[,<Size>]]

Related Commands	None
Arguments	<wfm_name>::=<string> is the waveform's name in waveform list. <StartIndex> ::= <NR1> specifies the start byte <Size> ::= <NR1> specifies the number of bytes to return.
Returns	<block_data>::=IEEE 488.2 block data format
Example	WLISt:WAVeform:DATA? "TestWfm",0,1024 returns #41024XXXX… where XXXX… are the first 1024 bytes of the waveform called "TestWfm".

Table 162: WLISt:WAVeform:DATA

Command	WLISt:WAVeform:DELete (No Query Form)
Description	This command deletes a specified waveform (or all waveforms) from the currently waveform list. It possible delete only the user-created waveforms. The operation is executed on current configuration only. If the deleted waveform is currently loaded into waveform memory, it is unloaded. **Important Note: w**hen ALL is specified, all deletable waveforms in the database will be deleted in a single action. Note that there is no "UNDO" action once the waveforms are deleted. Use caution before issuing this command. Note: The AWG must be in idle state.
Group	Waveform
Syntax	WLISt:WAVeform:DELete {<wfm_name> | ALL}
Related Commands	WLISt:SIZE? WLISt:NAME?
Arguments	<wfm_name>::=<string>
Returns	None
Example	WLISt:WAVeform:DELete ALL Deletes all user-created waveforms from the waveform list. The ALL parameter does not delete predefined waveforms. WLIST: WAVeform:DELETE "Test1" Deletes a waveform called "Test1".

Table 163: WLISt:WAVeform:DELete

Command	WLISt:WAVeform:IMPort

Description	This command imports the waveform from internal driver or USB driver into the waveform list. File formats supported: TXT: a list of numbers TRC: Lecroy format ZIP: file archive Active Technologies format. **NOTE 1**: If waveform file has ".zip" extension then the parameter {ANAlog | DIGitals} will not be taken into account. The ".zip" format already has this information in itself. **NOTE 2**: If the waveform name is already present in the waveform list then an error will occur. **NOTE 3**: If the waveform type {ANAlog | DIGitals} is not specified, the ANAlog type will be assumed as default. **NOTE 4:** Only removable units and "C:\Users\<username>\Pictures\Saved_Pictures" directory are accessible by MMEMory commands. The waveform path may contain a full file path. However, if the file path only contains a waveform name, the waveform will be searched starting from the current directory. **NOTE 5:** This operation is equivalent to what the user can do through the following user interface menu: Wave. List -> Import
Group	Waveform
Syntax	WLISt:WAVeform:IMPort <wfm_name>,<file_name>[,{ANAlog | DIGitals}]
Related Commands	WLISt:SIZE? WLISt:NAME? WLISt:WAVeform:DELete
Arguments	<wfm_name>::=<string> <file_name>::=<string>.{zip | trc | txt} indicates the absolute or relative path of the waveform file to import. ANAlog: the waveform is imported as analog waveform DIGitals: the waveform is imported as digitals waveform
Returns	None

Example	WLIST:WAVEFORM:IMPort "MyImportedWave", "E:/WaveLib/Test1.txt", ANAlog Imports a waveform called "Test1.txt" located in E:/WaveLib directory in waveform list; it will be called "MyImportedWave" and it will appear as analog waveform.

Table 164: WLISt:WAVeform:IMPort

Command	WLISt:WAVeform:LMAXimum (Query Only)
Description	This command returns the maximum number of waveform sample points allowed. The returned value is dependent on the instrument model and the installed options.
Group	Waveform
Syntax	WLISt:WAVeform:LMAXimum?
Related Commands	WLISt:WAVeform:LMINimum?
Arguments	None
Returns	<NR1>
Example	WLISt:WAVeform:LMAXimum? Returns 1073741428 (if 1G option is installed)

Table 165: WLISt:WAVeform:LMAXimum

Command	WLISt:WAVeform:LMINimum (Query Only)
Description	This command returns the minimum number of waveform sample points required for a valid waveform.
Group	Waveform
Syntax	WLISt:WAVeform:LMINimum?
Related Commands	WLISt:WAVeform:LMAXimum?
Arguments	None
Returns	<NR1>
Example	WLISt:WAVeform:LMINimum? Returns 16

Table 166: WLISt:WAVeform:LMINimum

Command	WLISt:WAVeform:LENGth (Query Only)
Description	This query returns the size of the waveform. The returned value represents data points (not bytes).

Group	Waveform
Syntax	WLISt:WAVeform:LENGth? <wfm_name>
Related Commands	None
Arguments	<wfm_name>::=<string>
Returns	<NR1>
Example	WLISt:WAVeform:LENGth? "Sine_360samples" Might return 360 indicating that the waveform contains 360 samples.

Table 167: WLISt:WAVeform:LENGth

Command	WLISt:WAVeform:PREDefined (Query Only)
Description	This query returns true or false based on whether the waveform is predefined. **Important Note:** Predefined waveforms have fixed length and name. Therefore, renaming or deleting them is not possible. Creating a new waveform with the same name as the predefined waveform is not possible.
Group	Waveform
Syntax	WLISt:WAVeform:PREDefined? <wfm_name>
Related Commands	None
Arguments	<wfm_name>::=<string>
Returns	0|1 where '0' means FALSE and '1' means TRUE
Example	WLISt:WAVeform:PREDefined? "Sine" might return 1 indicating that it is a predefined waveform.

Table 168: WLISt:WAVeform:PREDefined

Command	WLISt:WAVeform:TYPE (Query Only)
Description	This query returns if the waveform is analog or digital.
Group	Waveform
Syntax	WLISt:WAVeform:TYPE? <wfm_name>
Related Commands	None
Arguments	<wfm_name>::=<string>
Returns	DIGital|ANAlog
Example	WLISt:WAVeform:TYPE? "Ramp1000" Might return ANAlog.

Table 169: WLISt:WAVeform:TYPE

2.25 Multi Instrument Group Commands

The multi instrument synchronization is available on 8 channel models only.
The following commands have effect on 8 channel models only.

Command	MIM:CAPTure (No Query Form)
Description	This command captures all slave instruments connected to the master in which the command is sent.
Group	Multi Instrument
Syntax	MIM:CAPTure
Related Commands	MIM:RELease
Arguments	NONE
Returns	
Example	MIM:CAPTure This command sent on a master instrument capture the connected slave instruments.

Table 170: MIM:CAPTure

Command	MIM:ID (Query Only)
Description	This query command allows to identify the instrument of a chain. The instrument returns its number in the chain starting from the master that is 0.
Group	Multi Instrument
Syntax	MIM:ID?
Related Commands	NONE
Arguments	NONE
Returns	<NR1>
Example	MIM:ID? 1 indicates the first slave of the chain.

Table 171: MIM:ID

Command	MIM:CAPTured (Query Only)
Description	This query return whether an instrument is captured by a master
Group	Multi Instrument
Syntax	MIM:CAPTured?
Related Commands	NONE
Arguments	NONE

Returns	0|1 where '0' means FALSE and '1' means TRUE
Example	MIM:CAPTured?Might return 1. 1 indicates that the instrument has been captured by a master.

Table 172: MIM:CAPTured

Command	MIM:FORWard (Query Only)
Description	This query returns whether another instrument is connected to the "Synch Out" port of this instrument.
Group	Multi Instrument
Syntax	MIM:FORWard?
Related Commands	NONE
Arguments	NONE
Returns	0|1 where '0' means FALSE and '1' means TRUE
Example	MIM:FORWard? Might return 1. 1 indicates that another instrument is present and ready to be captured.

Table 173: MIM:FORWard

Command	MIM:SLAve (Query Only)
Description	This query returns wherther an instrument is slave or master. If it returns 1 it indicates that there is another instrument connected to the "Sync In" port, but it doesn't mean that the slave is captured.
Group	Multi Instrument
Syntax	MIM:SLAve?
Related Commands	NONE
Arguments	NONE
Returns	0|1 where '0' means FALSE and '1' means TRUE
Example	MIM:SLAve? Might return 1. 1 indicates that the instrument has another instrument connected to the "Sync In" port.

Table 174: MIM:SLAve

Command	MIM:NUMber (Query Only)
Description	This query returns the number of captured intruments.
Group	Multi Instrument

Syntax	MIM:NUMber?
Related Commands	None
Arguments	None
Returns	<value> ::=<NR1> is the number of captured instrument.
Example	MIM:NUMber? Might return 2. That indicates that the instrument is master and it captured 2 other instruments.

Table 175: MIM:NUMber

Command	MIM:RELease (No Query Form)
Description	This command sent on the master of a chain releases all captured instruments.
Group	Multi Instrument
Syntax	MIM:RELease
Related Commands	MIM:CAPTure
Arguments	None
Returns	None
Example	MIM:RELease This command sent on a master instrument release the captured slave instruments.

Table 176: MIM:RELease

3. COMMAND ERRORS

Command errors are returned when there is a syntax error in the command.

Error code	Error message
-399	Queue Overflow
-398	System Error
-397	No channels available
-396	Error to parsing waveform block data
-395	Waveform not found or not defined
-394	Sequencer not defined or incorrect data
-393	Waveform definition error
-392	Waveform out of range
-391	License option error

-390	Invalid run mode
-389	Subsequence error
-388	File Error
-387	Out of range error
-386	Application error
-385	Data type error
-384	Diagnostic error
-383	Calibration error
-382	Error on loading setting into the instrument
-381	Error on writing the serial number
-380	Error on reading the status of the digital input pins
-379	Waveform length different between channels in one sequencer/subsequencer entry
-378	Unable to set the Attenuation Value; if the Attenuation value is greater than 20dB, the Vocm value must be in the following range: +-250mV
-377	Predefined Waveform Error (it is not possible to delete or create a predefined waveform)
-376	Waveform Granularity Error
-375	Digital port disabled error: the digital port is disabled.
-374	The start operation is failed
-373	The stop operation is failed
-372	The trigger operation is failed
-371	Load configuration error
-370	Import configuration error
-369	Save configuration error
-368	A configuration with the same name is already present
-367	Delete configuration error
-366	The configuration is locked
-365	The parameter is disabled
-364	No digitals programmed
-363	Wrong format
-362	The device is in running status
-361	Load license error
-360	Lock unlock license error
-359	Copy configuration error

-358	Delete waveform error
-357	No valid MSUS
-356	No valid folder
-355	No valid file
-354	IO file error
-353	File system operation error
-352	Start index error
-351	Size error
-350	Load waveform error
-349	Waveform already present error
-348	Wrong path error
-347	Predefined waveform
-346	Wrong block data
-345	Sequencer memory overflow
-344	No license installed
-343	Wrong waveform name
-342	The device is captured from the master device
-341	Multi Instrument management error
0	No error
5	DESIGN ERROR: Too many numeric suffices in Command Spec
10	No Input Command to parse
14	Numeric suffix is invalid value
16	Invalid value in numeric or channel list, e.g. out of range
17	Invalid number of dimensions in a channel list
20	Parameter of type Numeric Value overflowed its storage
30	Wrong units for parameter
40	Wrong type of parameter(s)
50	Wrong number of parameters
60	Unmatched quotation mark (single/double) in parameters
65	Unmatched bracket
70	Command keywords were not recognized
200	No entry in list to retrieve (number list or channel list)

210	Too many dimensions in entry to be returned in parameters
220	; plus End of line commands

Table 177: Command Errors

4. PREDEFINED WAVEFORMS

SINE	Sine waveform, 16384 samples, analog, normalized
RAMP	Ramp waveform,16384 samples, analog, normalized
SQUARE	Square waveform,16384 samples, analog, normalized
SYNC	Sync waveform,16384 samples, analog, normalized
DC	DC level waveform,16384 samples, analog, normalized
GAUSSIAN	Gaussian waveform,16384 samples, analog, normalized
LORENTZ	Lorentz waveform,16384 samples, analog, normalized
HAVERSINE	Haversine waveform,16384 samples, analog, normalized
EXP_RISE	Exponential rise waveform,16384 samples, analog, normalized
EXP_DECAY	Exponential decay waveform,16384 samples, analog, normalized
ZERO	All Zeros,16384 samples, digital
ONE	All Ones,16384 samples, digital
COUNTER	Counter,256 samples, digital
CLOCK	Clock,16 samples, digital

Table 178: Predefined Waveforms

5. REMOTE CONTROL

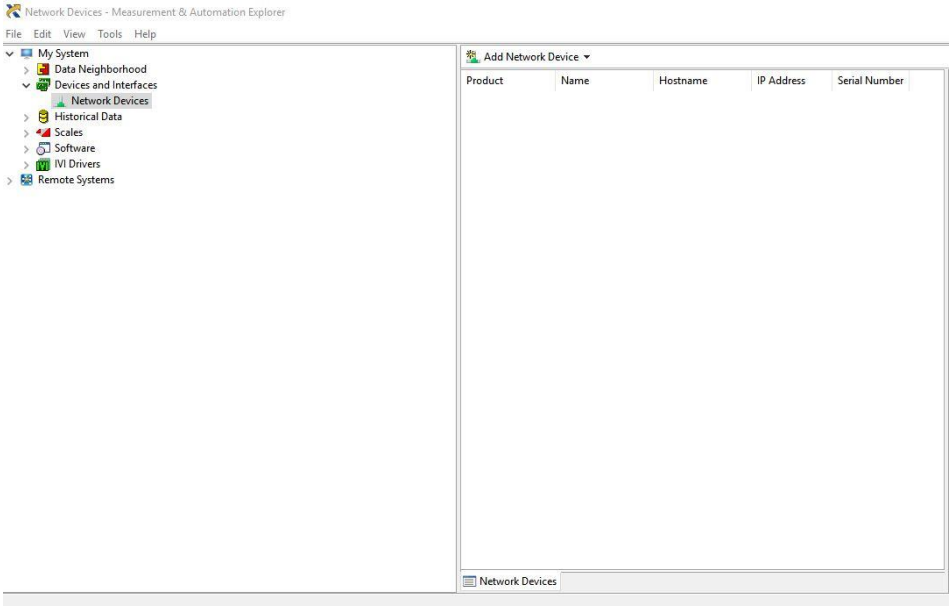
You can connect your instrument to a network for printing, file sharing, and Internet access, among other functions. Consult with your network administrator and use the standard Windows utilities to configure the instrument for your network.
The instrument can be controlled using VXI-11 (LAN) protocol. It allows you to control the instrument remotely by using SCPI commands. Please refer to the Model 675 programmer manual for a complete description about all available commands. You can follow the next steps to communicate with your Model 675 Series instrument:

5.1 Prerequisite

NI-VISA
VISA provides the programming interface between the hardware and development environments such as Visual Studio .NET, LabVIEW, LabWindows/CVI, Measurement Studio for Microsoft Visual Studio and MatLab. NI-VISA is the National Instruments implementation of the VISA I/O standard. NI-VISA includes software libraries, interactive utilities such as NI I/O Trace and the VISA Interactive Control, and configuration programs through Measurement & Automation Explorer for all your development needs.

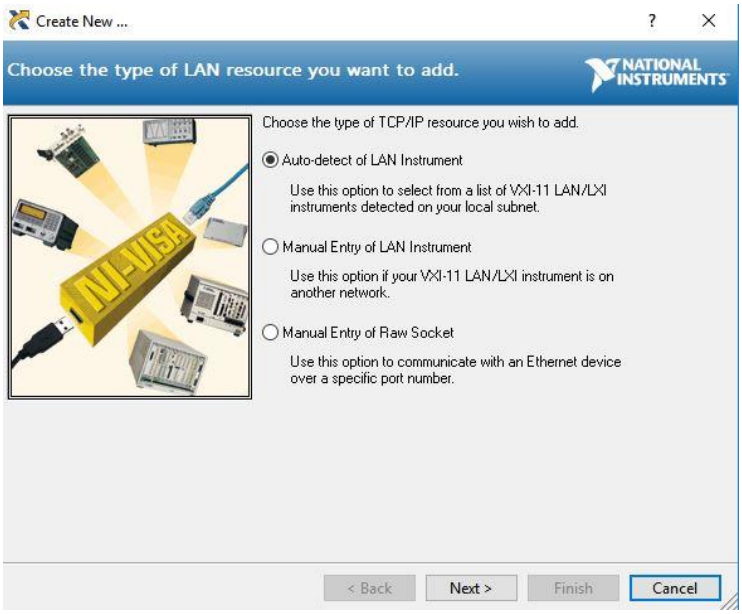
1. Connect your LAN cable to the instrument.
2. On the Client-PC you must install the latest NIVISA package that you can find here https://www.ni.com/it-it/support/downloads/drivers/download.ni-visa.html

3. Launch the NI-MAX tool on the Client-PC

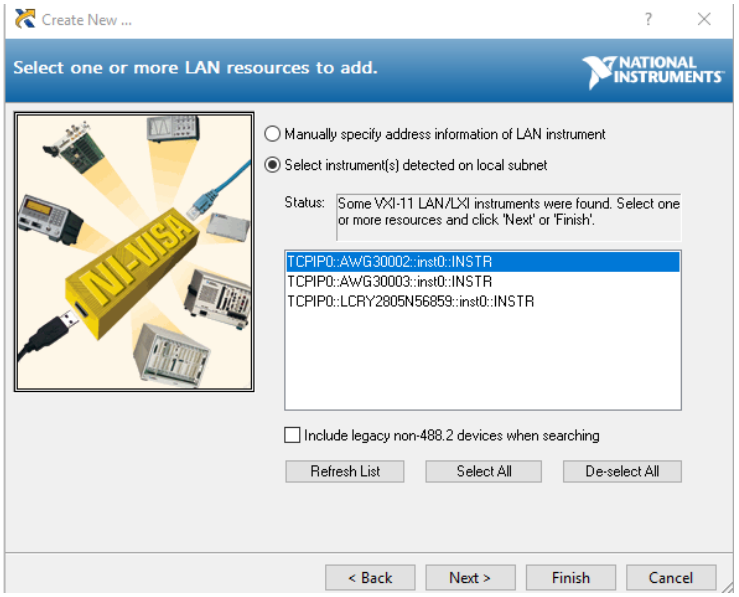


Press Add Network Device → VISA TCP/IP Resource…

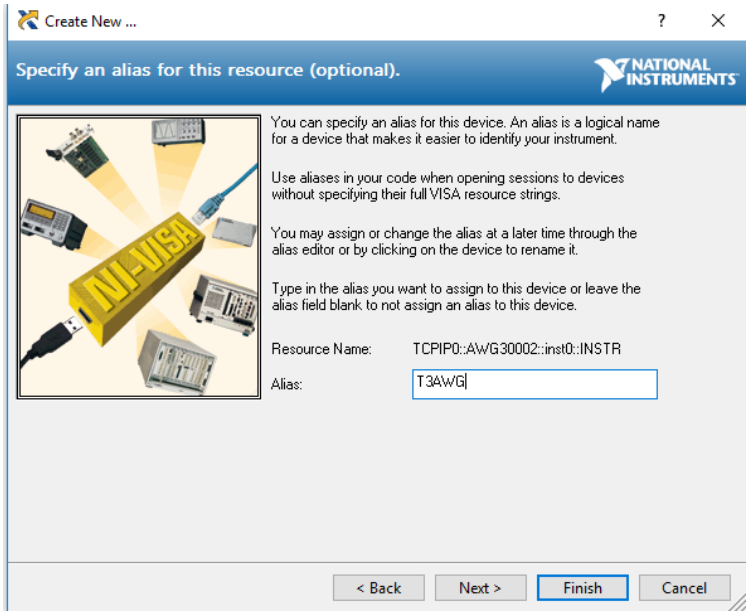
4. Select Auto-detect of LAN Instrument



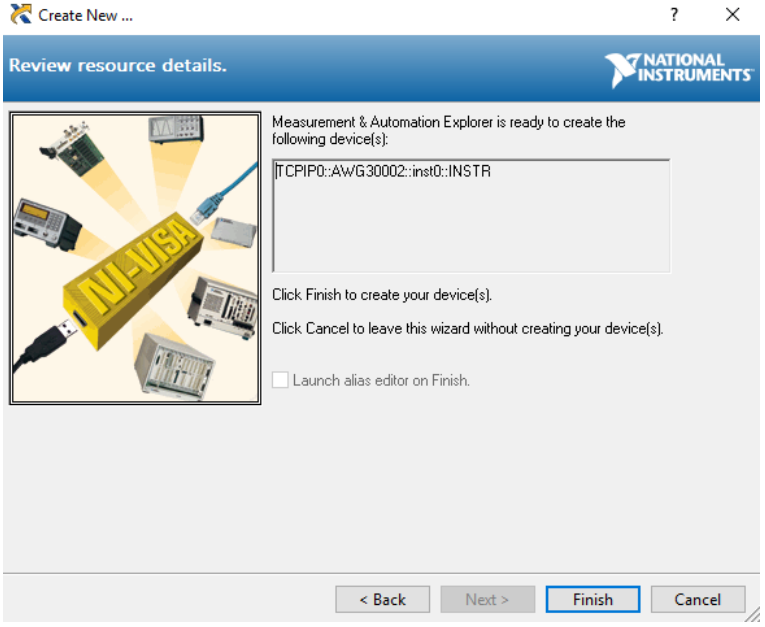
5. The panel will retrieve the discovered instruments on the LAN network, you should select the AWG4010X series one.



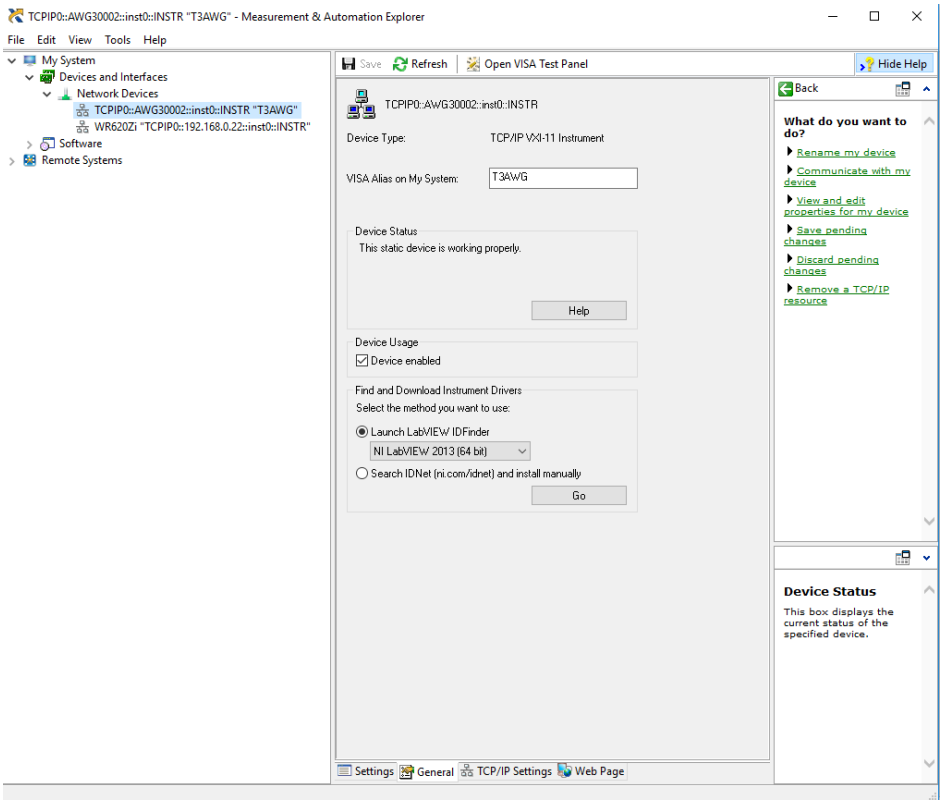
6. Specify an Alias for the selected resource



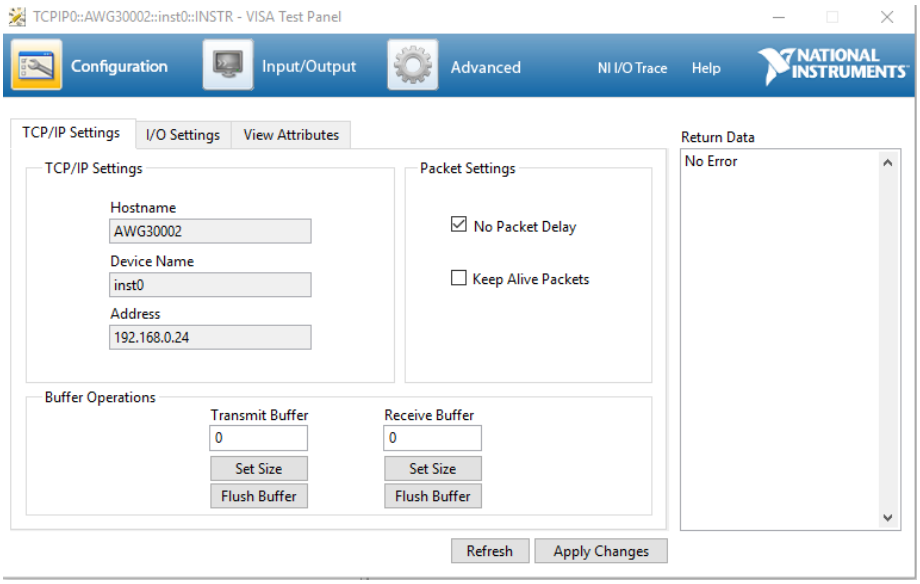
7. Press Finish



8. The AWG resource will be available in the Network Devices list

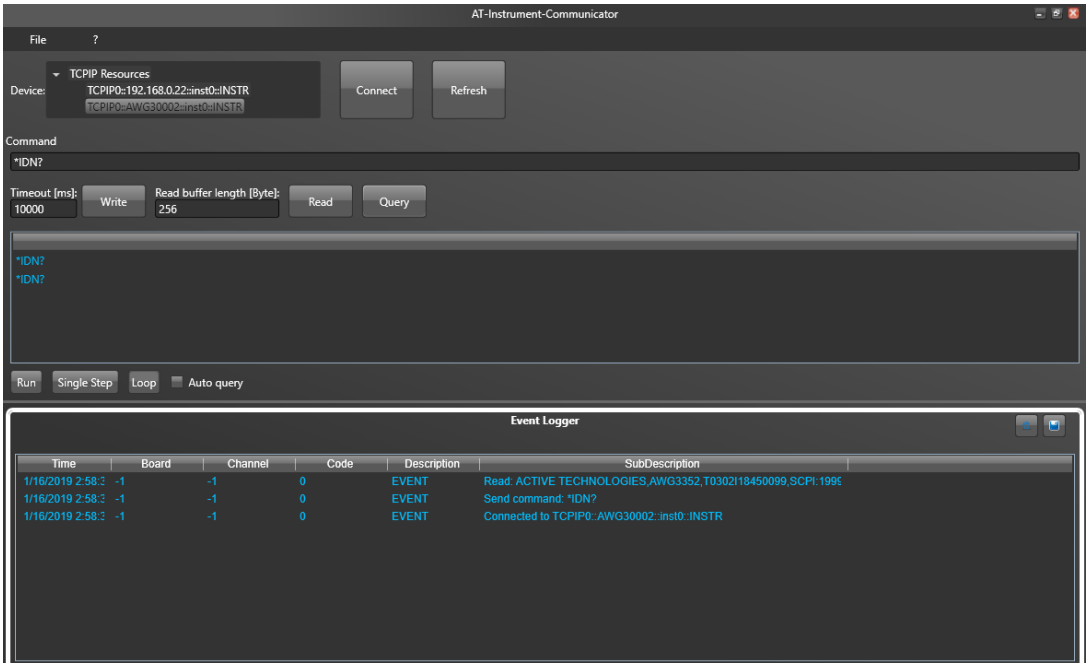


9. Now you can use send the SCPI commands to the AWG resource using the NI Visa Test Panel or the AT-Instrument-Communicator



10. On the Client-PC (IP Address) or AWG instrument (LocalHost), launch the AT-Instrument-Communicator tool

5.1.1 AT Instrument Communicator



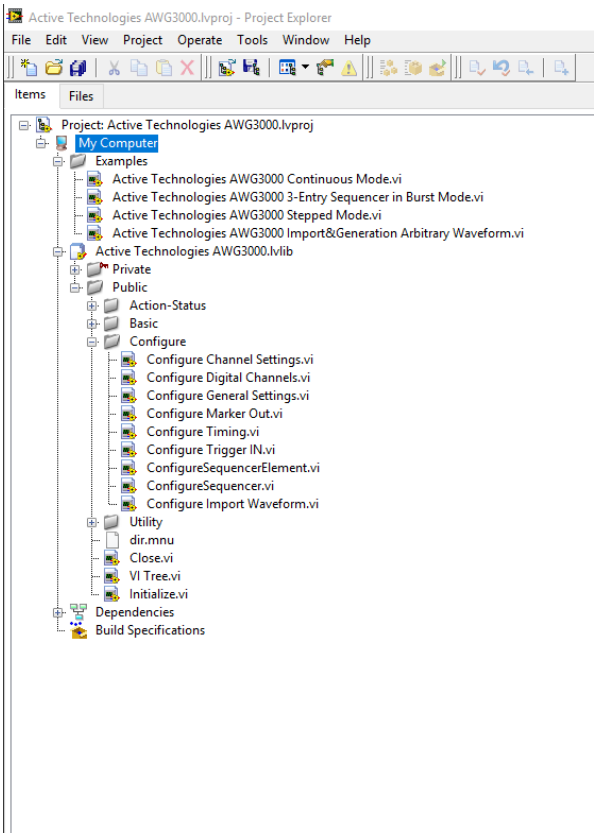
The AT-Instrument-Communicator software is a client-side component tool that uses NI-VISA on each remote PC, you must install a copy of NIVISA to make use of this client-side component (please follow the Prerequisite steps).

1. On the Client-PC launch the *AT Instrument Communicator* setup you can find in the folder "SDK_TRUEARB_4010" and install the software.
2. Select the AWG4010X resource on the Device list
3. Press the Connect button
4. If the instrument connection will be established, the SCPI command button will be enabled.
5. Write *IDN? in the command
6. Press the Query button

7. In the Event Logger list, the instrument should respond like this: ACTIVE TECHNOLOGIES,AWG4012,T0302I000001,SCPI 99.0,SV 1.0.34.0 where T0302I000001 is the serial number, SCPI 99.0 is the SCPI command version and SV 1.0.34.0 is the Software Version.
8. A command script is a list of SCPI commands (one command for each line) saved in a txt file; you can send a command script using the File → Load Script menu item.

5.2 NI LabView Examples

The LabView examples require at least LabView 2013 64 bit version, you should copy SDK_TRUEARB_4010 folder in …\LabVIEW 2013\instr.lib folder on your computer and open the file Active Technologies AWG3000.lvproj.



The LabView project contains several Vis that control the basic instrument features and four examples located in the folder *Examples*.

Note: for proper usage of Labview's Vis it's necessary that decimal symbol separator is properly set on your PC.
So please check that the Dot symbol "." is set in *Decimal Symbol* field that you can find:

In Windows 10

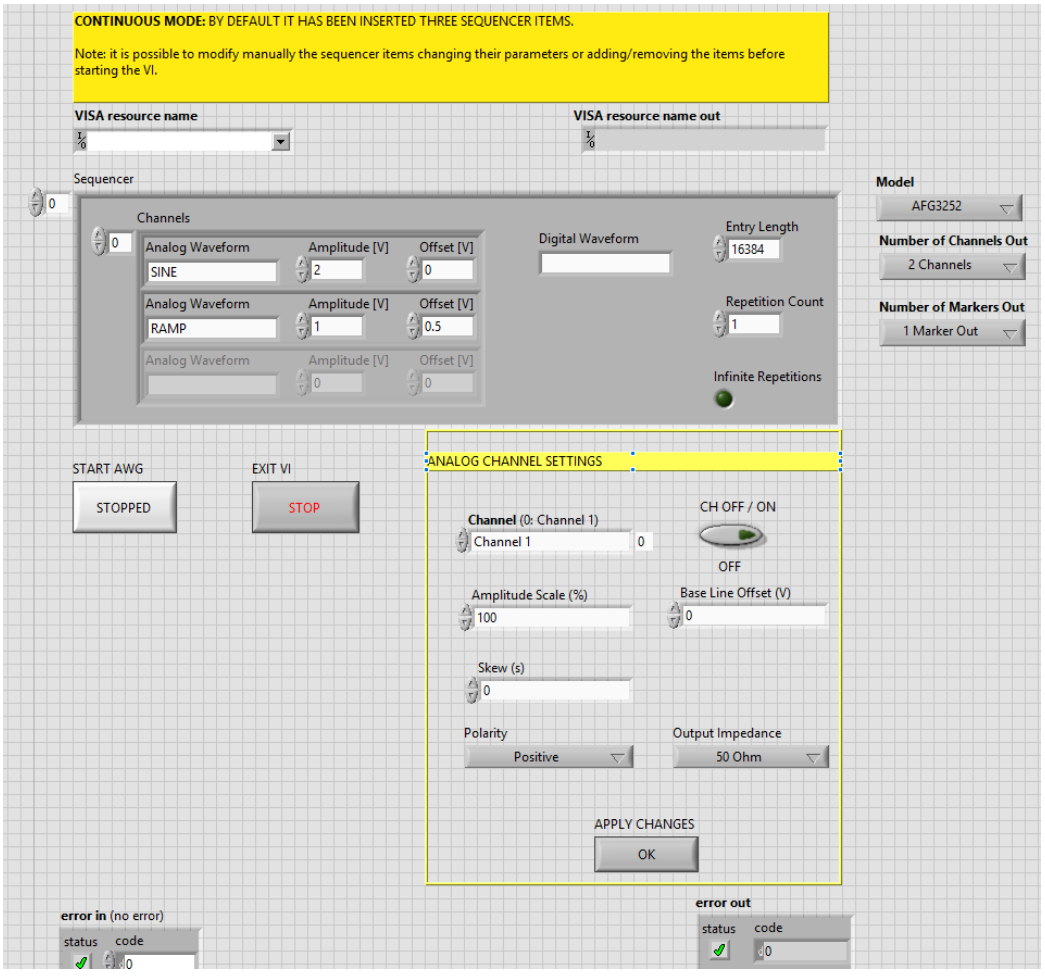
1. From the **Start** button click **Control Panel**.
2. Click **Region**.
3. On the **Formats** tab click the **Additional Settings** button.

In Windows 7

1. From the **Start** button click **Control Panel**.
2. Click **Regional and Language Options**.
3. On the **Formats** tab, under **Current format**, click **Customize this format**.

5.2.1 Continuous Mode

Double click on the project tree to launch the *Active Technologies AWG3000 Continuous Mode.vi* example



This example generates a sequence of waveforms in Continuous mode; the sequencer by default is made of three entries and it is loaded as follows:

	Entry 1 – Length 16384, Rep. 1	**Entry 2 – Length 16384, Rep. 2**	**Entry 3 – Length 16384, Rep. 3**
CH1	Sine, Amp. 2V, Offset 0V	Ramp, Amp. 3V, Offset 0V	Lorentz, Amp. 4V, Offset 0V
CH2	Ramp, Amp. 1V, Offset 0.5V	Sync, Amp. 3V, Offset 0V	Lorentz, Amp. 4V, Offset 0V

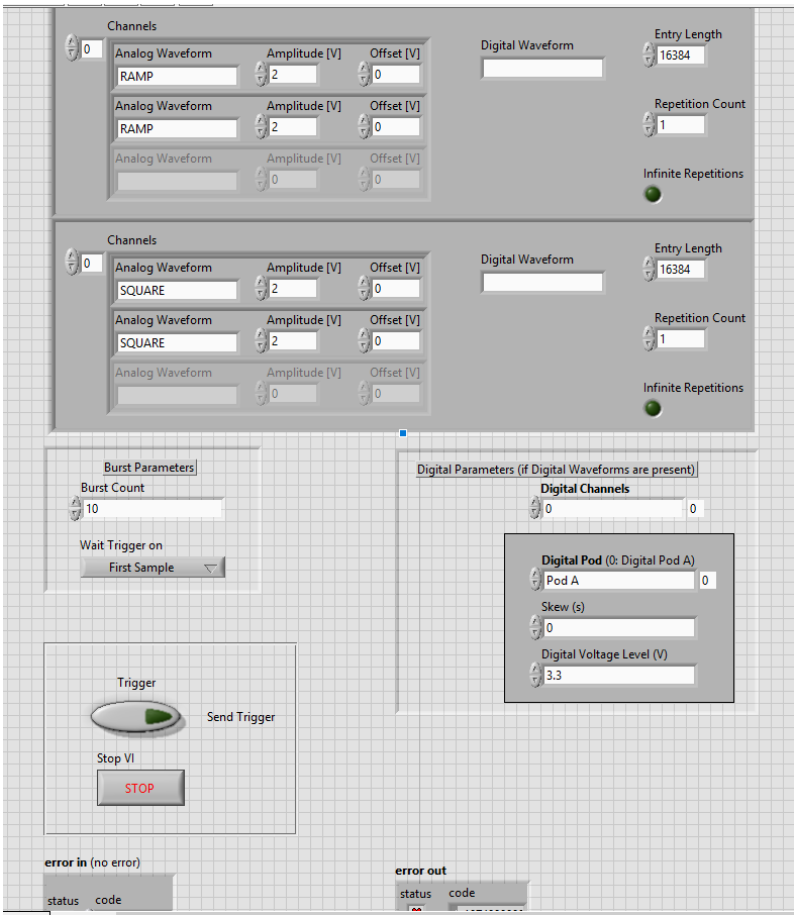
Before running the VI you have to select the AWG-4010 resource in the VISA resource name control.

- Run the VI and press the "START AWG" button to start the generation.
- The Analog Channel Settings section lets you to change on the fly the Amplitude Scale, the Base Line Offset, the Skew, the Polarity and the Output Impedace parameters. Select the Channel and press the Apply Changes button to confirm the changes.
- You can press the CH OFF/ON button to enable or disable the analog channel.

The entire sequence will be repeated continuously until you press again the "START AWG" button.

5.2.2 Burst Mode

Double click on the project tree to launch the *Active Technologies AWG3000 3-Entry Sequencer in Burst mode.vi* example



This example generates a sequence of waveforms in Burst Mode.
The sequencer by default is loaded as follows:

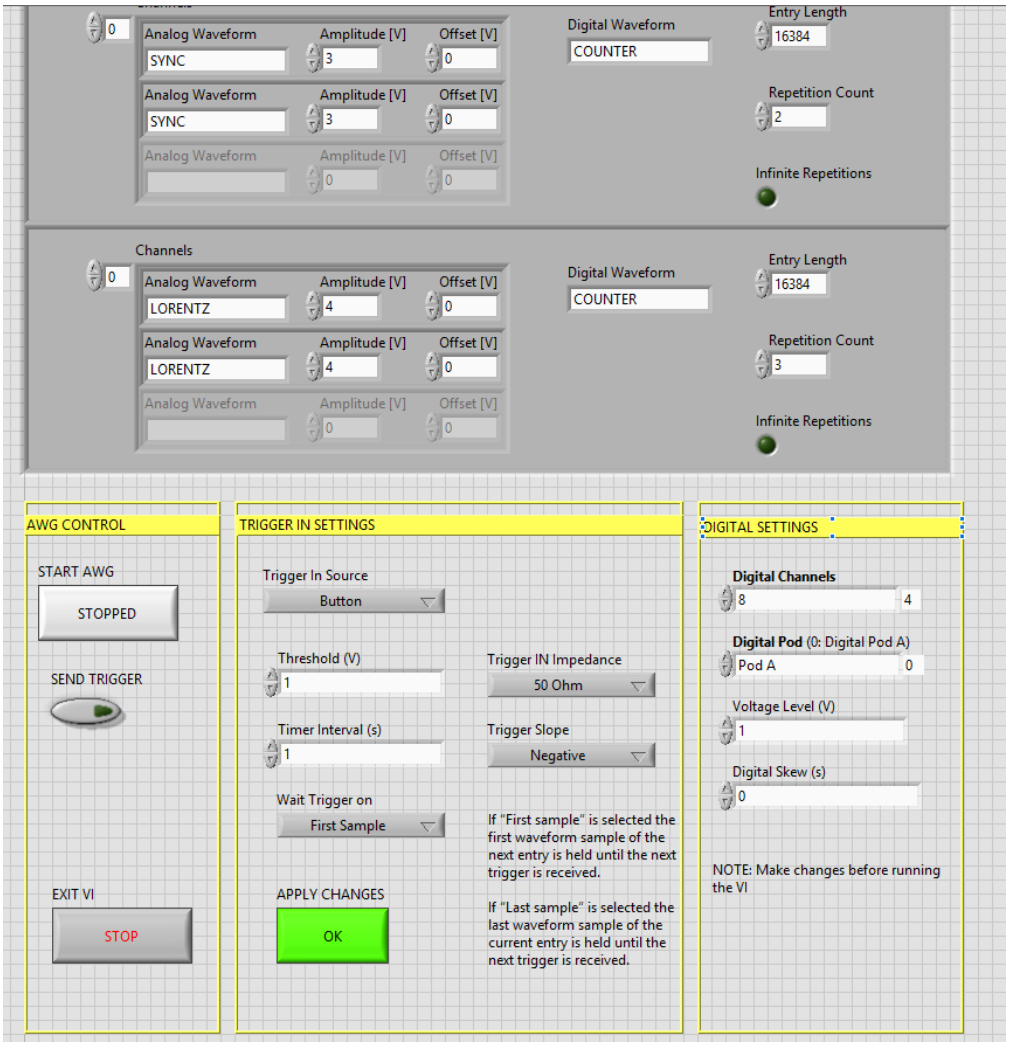
	Entry 1 – Length 16384, Rep. 1	**Entry 2 – Length 16384, Rep. 1**	**Entry 3 – Length 16384, Rep. 1**
CH1	Sine, Amp. 2V, Offset 0V	Ramp, Amp. 2V, Offset 0V	Square, Amp. 2V, Offset 0V
CH2	Sine, Amp. 2V, Offset 0V	Ramp, Amp. 2V, Offset 0V	Square, Amp. 2V, Offset 0V

In the "Burst Parameters" section, you can change the number of burst (Burst Count) and the Wait Trigger On parameter.
If you load Digital Waveforms in the sequencer, in the "Digital Parameters" section, you can select the number of Digital channels, the Digital Pod, the Skew and the Digital Voltage Level.

Run the VI to initialize the instrument and load the default parameters into the instrument; the Send Trigger button starts the waveform sequence burst.
Press the STOP button to stop the waveform generation and the VI.

5.2.3 Stepped Mode

Double click on the project tree to launch the *Active Technologies AWG3000 Stepped Mode.vi* example



This example generates a sequence of waveforms in Stepped Mode.
In this mode after pressing the RUN/STOP button each entry waits for a trigger event before its execution. The waveform of the entry will loop as written in the entry repetition parameter.

The sequencer by default is loaded as follows:

	Entry 1 – Length 16384, Rep. 1	**Entry 2 – Length 16384, Rep. 2**	**Entry 3 – Length 16384, Rep. 3**
CH1	Sine, Amp. 2V, Offset 0V	Sync, Amp. 3V, Offset 0V	Lorentz, Amp. 4V, Offset 0V

CH2	Exp. Rise, Amp. 2V, Offset 0V	Sync, Amp. 3V, Offset 0V	Lorentz, Amp. 4V, Offset 0V
DIG. WAV.	COUNTER	COUNTER	COUNTER

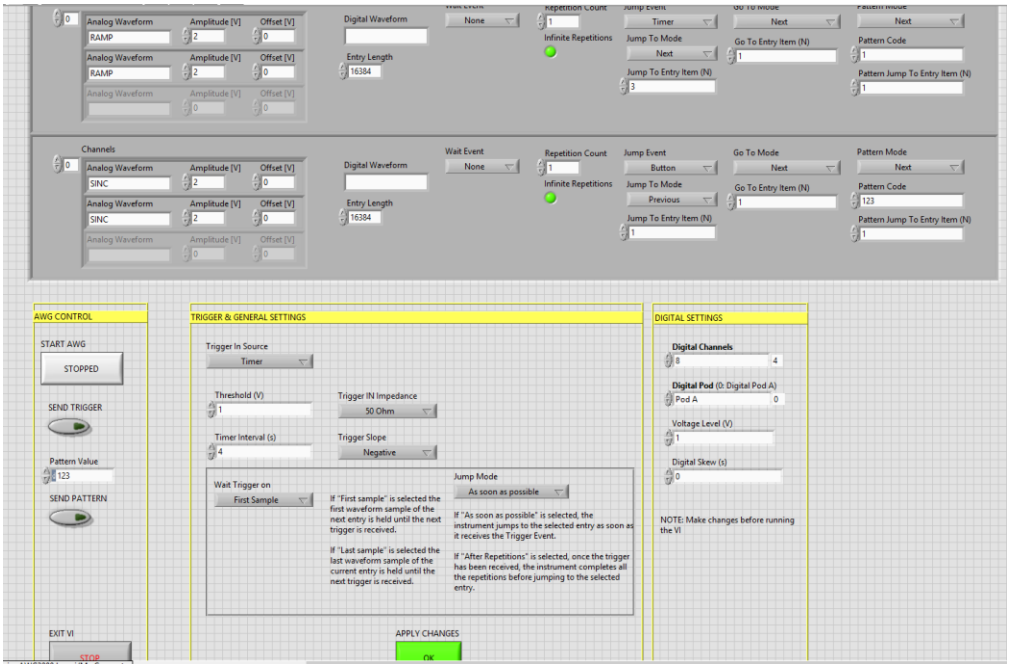
Before running the VI you can configure in the "Digital Settings" the number of digital channels, the digital pod voltage level and the digital skew.

1. Run the VI: by default the instrument will receive the trigger from the front panel button
2. Press the "START AWG" button to run the AWG
3. Press the "SEND TRIGGER" button to send a software trigger

In the "Trigger IN Settings", you can configure the Trigger In Source, the Threshold, the Trigger IN impedance, the Timer Interval, the Trigger Slope and the Wait Trigger On Parameter.
Stop the instrument and then press the "APPLY CHANGES" button to confirm changes on the TRIGGER IN settings.

5.2.4 Advanced Mode

Double click on the project tree to launch the *Active Technologies AWG3000 Advanced Mode.vi* example



This example generates a sequence of waveforms in Advanced Mode.
In this mode the execution of the sequence can be changed by using conditional and unconditional jumps (JUMPTO and GOTO commands) and dynamic jumps (PATTERN JUMP commands).

The sequencer by default is loaded as follows:

	Entry 1 – Length 16384 - Wait Event Button | Rep. 100 | Infinite Repetitions Off | Jump Event None | Jump To Mode Next | Jump to Entry Item 1 | Go To Mode Next | Go To Entry Item 1 | Pattern Mode Next | Pattern Code 1 | Pattern Jump To Entry Item 1	**Entry 2 – Length 16384 - Wait Event None | Rep. 1 | Infinite Repetitions On | Jump Event Timer | Jump To Mode Next | Jump to Entry Item 3 | Go To Mode Next | Go To Entry Item 1 | Pattern Mode Next | Pattern Code 1 | Pattern Jump To Entry Item 1**	**Entry 3 – Length 16384 – Wait Event None | Rep. 1 | Infinite Repetitions On | Jump Event Button | Jump To Mode Previous | Jump to Entry Item 1 | Go To Mode Next | Go To Entry Item 1 | Pattern Mode Next | Pattern Code 123 | Pattern Jump To Entry Item 1**
CH1	Sine, Amp. 2V, Offset 0V	Ramp, Amp. 2V, Offset 0V	Sinc, Amp. 2V, Offset 0V
CH2	Sine, Amp. 2V, Offset 0V	Ramp, Amp. 2V, Offset 0V	Sinc, Amp. 2V, Offset 0V

Before running the VI you can configure in the "Digital Settings" the number of digital channels, the digital pod voltage level and the digital skew. You can also add a Digital Waveform to the sequencer.

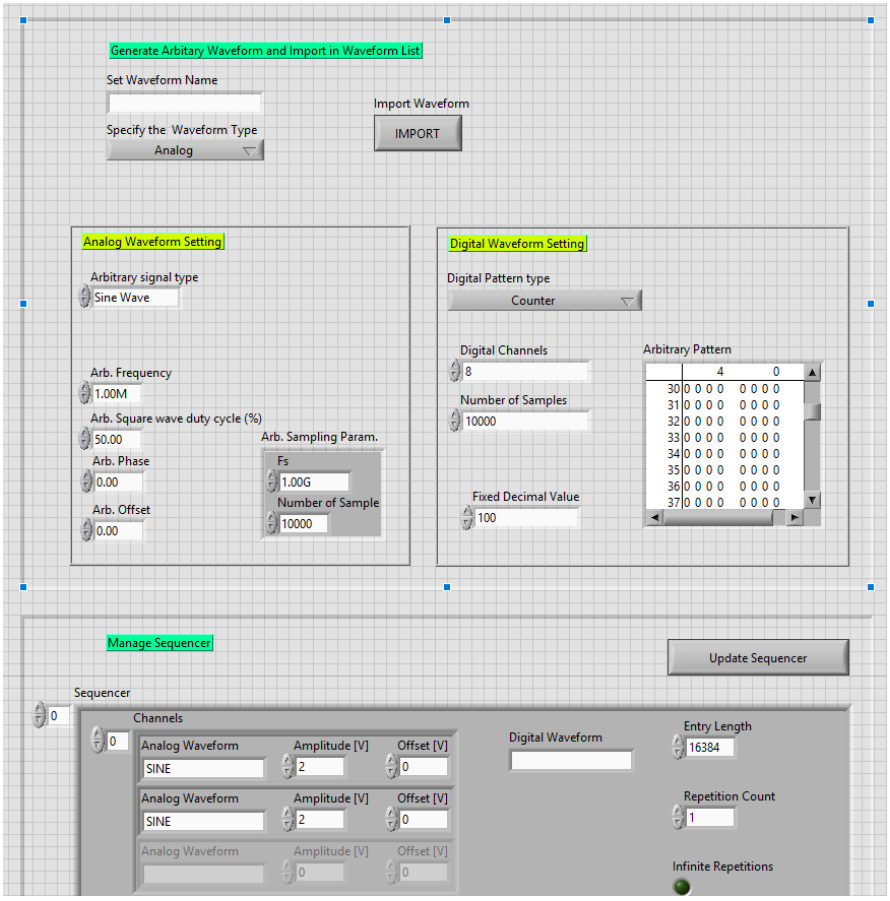
4. Run the VI. By default the instrument will receive the trigger from the front panel button.
5. Press the "START AWG" button to run the AWG.
6. You can press the "SEND TRIGGER" button to send a software trigger, you can press the "SEND PATTERN" button to send the pattern code value.
7. By default the execution of the sequencer is the following one: press the SEND TRIGGER button to start the waveform repetitions of the first entry.

8. On the second entry, the instrument waits for the timer event before executing the JUMP to the third entry.
9. On the third entry, the instrument can accept the trigger button (software trigger) as Jump Event to the Previous Entry or the PATTERN code value to Jump to the first entry of the sequencer: you can press the SEND TRIGGER button or the SEND PATTERN button to change the execution order of the waveform sequence.

In the "Trigger & General Settings", you can configure the Trigger In Source, the Threshold, the Trigger IN impedance, the Timer Interval, the Trigger Slope, the Wait Trigger On Parameter and the Jump Mode parameter.
Stop the instrument and then press the "APPLY CHANGES" button to confirm changes on the TRIGGER & GENERAL settings.

5.2.5 Import an Arbitrary Waveform Generation

This example lets you to create analog or digital waveforms, import them in the Waveform List and then load the sequencer with the generated waveforms. By default the instrument is set in Continuous mode.

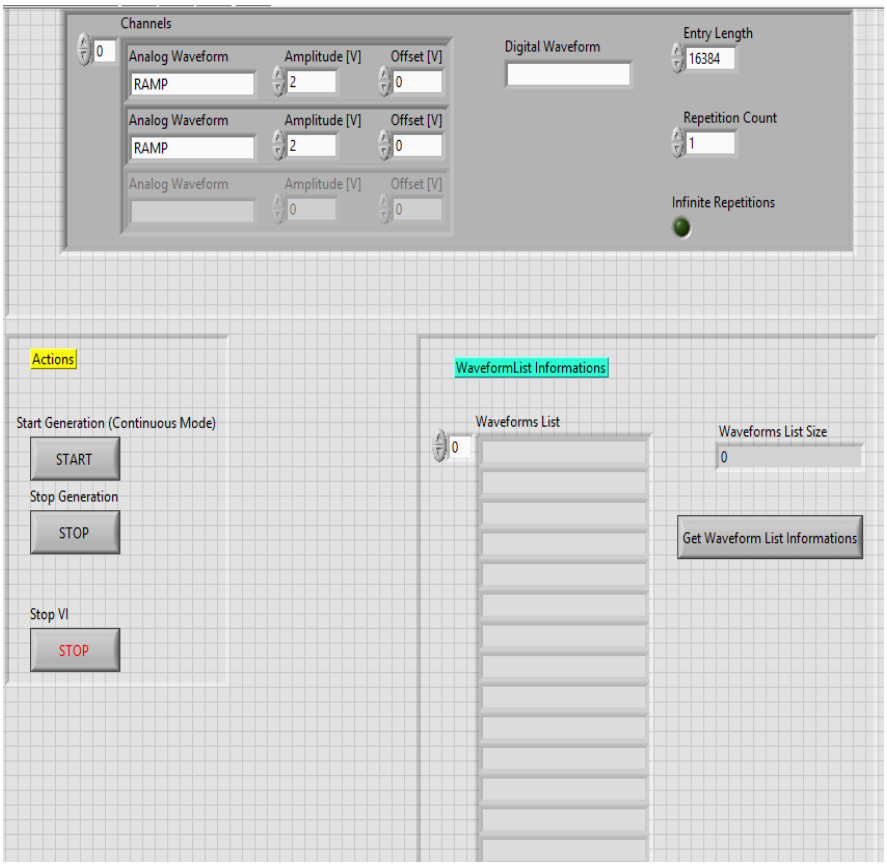


In the Analog Waveform Setting section you have different parameters to set the analog waveform: you can select the arbitrary signal type between Sine, Triangle, Square and Sawtooth waveform, you can change the frequency, the duty cycle, the phase and the offset. Moreover you can set the sampling rate and the number of samples.

In the Digital Waveform Setting section you have different parameters to set the digital waveform; in the Digital Pattern type dropdown list you can select between a counter, a random number, a fixed value or an arbitrary pattern.
When you select "arbitrary pattern", you can set in the Arbitrary Pattern table the samples of the waveform: the row is the sample number and the column is the digital line.

The Digital Channels and the Number of Samples parameters define the number of digital lines available in the instrument and the number of samples of the digital waveform.

The Manage Sequencer section allows you to configure the sequencer with predefined or arbitrary waveforms.



The "WaveformList Informations" section allows you to know the names of the available predefined and imported waveforms in the instrument.

The following steps describe how to create and import an analog/digital waveform:

1. Run the VI
2. In the "Set Waveform Name" control write SINE_120MHZ
3. Set Analog in the "Specify the Waveform Type" dropdown list
4. In the Analog Waveform Setting section set Sine Wave as signal type, 120 MHz in the Arb. Frequency, 1 GHz as Fs (sampling rate) and 12000 as number of samples.
5. Press the Import button
6. In the "Set Waveform Name" control write RAND_12000
7. Set Digitals in the "Specify the Waveform Type" dropdown list
8. In the Digital Waveform Setting section select Random in the Digital Pattern Type, set 8 Digital Channels and 12000 in the Number of Samples Parameter
9. Press the Import button

10. Press the button "Get Waveform List Informations" and in the waveform list it will appear SINE_120MHZ and RAND_12000.
11. The sequencer by default will be loaded as:

	Entry 1 – Length 16384, Rep. 1	**Entry 2 – Length 16384, Rep. 1**	**Entry 3 – Length 16384, Rep. 1**
CH1	Sine, Amp. 2V, Offset 0V	Ramp, Amp. 2V, Offset 0V	Square, Amp. 2V, Offset 0V
CH2	Sine, Amp. 2V, Offset 0V	Ramp, Amp. 2V, Offset 0V	Square, Amp. 2V, Offset 0V
DIG.WAV.			

12. Replace the third entry with the imported ones

	Entry 1 – Length 16384, Rep. 1	**Entry 2 – Length 16384, Rep. 1**	**Entry 3 – Length 12000, Rep. 1**
CH1	Sine, Amp. 2V, Offset 0V	Ramp, Amp. 2V, Offset 0V	SINE_120MHZ, Amp. 2V, Offset 0V
CH2	Sine, Amp. 2V, Offset 0V	Ramp, Amp. 2V, Offset 0V	SINE_120MHZ, Amp. 2V, Offset 0V
DIG.WAV.	ONE	ZERO	RAND_12000

13. Press the Update Sequencer button to load the new sequencer
14. Press the START button to run the AWG

5.3 Script Examples

The scripts are contained in the folder “Sample Scripts”

5.3.1 Continuous Mode

```
*CLS
*IDN?
*RST
AWGControl:INCreasing INTERpolation
AWGControl:DECreasing DECIMation
AWGControl:RMODe CONTinuous
DISPlay:UNIT:VOLT AMPLitudeoff
SEQuence:LENGth 3
SEQuence:ELEM1:WAVeform1 "SINE"
SEQuence:ELEM1:AMPlitude1 2.000000
SEQuence:ELEM1:OFFset1 0.000000
SEQuence:ELEM1:WAVeform2 "RAMP"
SEQuence:ELEM1:AMPlitude2 1.000000
SEQuence:ELEM1:OFFset2 0.500000
SEQuence:ELEM1:LENGth 16384
SEQuence:ELEM1:LOOP:COUNt 1
SEQuence:ELEM2:WAVeform1 "RAMP"
SEQuence:ELEM2:AMPlitude1 3.000000
SEQuence:ELEM2:OFFset1 0.000000
SEQuence:ELEM2:WAVeform2 "SYNC"
SEQuence:ELEM2:AMPlitude2 3.000000
SEQuence:ELEM2:OFFset2 0.000000
SEQuence:ELEM2:LENGth 16384
SEQuence:ELEM2:LOOP:COUNt 2
SEQuence:ELEM3:WAVeform1 "LORENTZ"
SEQuence:ELEM3:AMPlitude1 4.000000
SEQuence:ELEM3:OFFset1 0.000000
SEQuence:ELEM3:WAVeform2 "LORENTZ"
SEQuence:ELEM3:AMPlitude2 4.000000
SEQuence:ELEM3:OFFset2 0.000000
SEQuence:ELEM3:LENGth 16384
SEQuence:ELEM3:LOOP:COUNt 3
AWGControl:CONFigure:CNUMber?
OUTPut1:STATe ON
OUTPut2:STATe ON
AWGControl:RUN
```

5.3.2 Stepped Mode

Note: the trigger is set as manual; this script includes also the digital lines

```
*CLS
*IDN?
*RST
AWGControl:CONFigure:CNUMber?
AWGControl:RMODe STEPped
AWGControl:WAITstate FIRST
DIGitals:NUMber 8
DIGitals:LEVel1 1.000000
DIGitals:SKEW1 0.000000
DIGitals:STATe ON
SEQuence:LENGth 3
DISPlay:UNIT:VOLT AMPLitudeoff
SEQuence:ELEM1:WAVeform1 "SINE"
SEQuence:ELEM1:AMPlitude1 2.000000
SEQuence:ELEM1:OFFset1 0.000000
SEQuence:ELEM1:WAVeform2 "EXP_RISE"
SEQuence:ELEM1:AMPlitude2 2.000000
SEQuence:ELEM1:OFFset2 0.000000
SEQuence:ELEM1:WAVeform3 "COUNTER"
SEQuence:ELEM1:LENGth 16384
SEQuence:ELEM1:LOOP:COUNt 1
SEQuence:ELEM2:WAVeform1 "SYNC"
SEQuence:ELEM2:AMPlitude1 3.000000
SEQuence:ELEM2:OFFset1 0.000000
SEQuence:ELEM2:WAVeform2 "SYNC"
SEQuence:ELEM2:AMPlitude2 3.000000
SEQuence:ELEM2:OFFset2 0.000000
SEQuence:ELEM2:WAVeform3 "COUNTER"
SEQuence:ELEM2:LENGth 16384
SEQuence:ELEM2:LOOP:COUNt 2
SEQuence:ELEM3:WAVeform1 "LORENTZ"
SEQuence:ELEM3:AMPlitude1 4.000000
SEQuence:ELEM3:OFFset1 0.000000
SEQuence:ELEM3:WAVeform2 "LORENTZ"
SEQuence:ELEM3:AMPlitude2 4.000000
SEQuence:ELEM3:OFFset2 0.000000
SEQuence:ELEM3:WAVeform3 "COUNTER"
SEQuence:ELEM3:LOOP:COUNt 3
TRIGger:SOURce MAN
```

```
OUTPut1:STATe ON
OUTPut2:STATe ON
AWGControl:RUN
*TRG
```

5.3.3 Import Arbitrary

1. Before running this script, please copy the folder "Example_Wave_TXT" into the instrument folder C:\Users\<USERNAME>\Pictures\Saved Pictures\
2. The 10000_Sample_Analog_Sine_Wave.txt contains a set of analog sine waves made of 10000 samples. The 10000_Sample_Digital_Random_Pattern.txt is a digital random pattern made of 10000 samples.
3. The aim of this example is to import them in the sequencer and run the instrument in Continuous mode.

```
*CLS
*IDN?
*RST
AWGControl:CONFigure:CNUMber?
AWGControl:RMODe CONTinuous
DISPlay:UNIT:VOLT AMPLitudeoff
DIGitals:NUMber 8
DIGitals:LEVel1 3.000000
DIGitals:SKEW1 0.000000
DIGitals:STATe ON
WLISt:WAVeform:IMPort "Sine_10000",
"\Example_Wave_TXT\10000_Sample_Analog_Sine_Wave.txt",ANAlog
WLISt:WAVeform:IMPort "Digital_10000",
"\Example_Wave_TXT\10000_Sample_Digital_Random_Pattern.txt",DIGital
SEQuence:LENGth 1
SEQuence:ELEM1:WAVeform1 "Sine_10000"
SEQuence:ELEM1:AMPlitude1 2.000000
SEQuence:ELEM1:OFFset1 0.000000
SEQuence:ELEM1:WAVeform2 "Sine_10000"
SEQuence:ELEM1:AMPlitude2 2.000000
SEQuence:ELEM1:OFFset2 0.000000
SEQuence:ELEM1:WAVeform3 "Digital_10000"
SEQuence:ELEM1:LENGth 10000
SEQuence:ELEM1:LOOP:COUNt 1
OUTPut1:STATe ON
OUTPut2:STATe ON
AWGControl:RUN
```

5.3.4 Advanced Mode

```
*CLS
*IDN?
*RST
AWGControl:INCreasing INTERpolation
AWGControl:DECreasing DECIMation
AWGControl:RMODe ADVAnced
AWGControl:WAITstate FIRST
AWGControl:JUMPMode IMMediate
AWGControl:CONFigure:CNUMber?
SEQuence:LENGth 3
DISPlay:UNIT:VOLT AMPLitudeoff
SEQuence:ELEM1:WAVeform1 "SINE"
SEQuence:ELEM1:AMPlitude1 2.000000
SEQuence:ELEM1:OFFset1 0.000000
SEQuence:ELEM1:WAVeform2 "SINE"
SEQuence:ELEM1:AMPlitude2 2.000000
SEQuence:ELEM1:OFFset2 0.000000
SEQuence:ELEM1:LENGth 16384
SEQuence:ELEM1:LOOP:COUNt 100
SEQuence:ELEM1:WAITEvent MANual
SEQuence:ELEM1:JUMPEvent NONE
SEQuence:ELEM1:JUMPTOMode NEXT
SEQuence:ELEM1:JUMPTOEntry 1
SEQuence:ELEM1:GOTOMode NEXT
SEQuence:ELEM1:GOTOEntry 1
SEQuence:ELEM1:PATTERNJUMPTOMode NEXT
SEQuence:ELEM1:PATTERN 1
SEQuence:ELEM1:PATTERNJUMPTOEntry 1
DISPlay:UNIT:VOLT AMPLitudeoff
SEQuence:ELEM2:WAVeform1 "RAMP"
SEQuence:ELEM2:AMPlitude1 2.000000
SEQuence:ELEM2:OFFset1 0.000000
SEQuence:ELEM2:WAVeform2 "RAMP"
SEQuence:ELEM2:AMPlitude2 2.000000
SEQuence:ELEM2:OFFset2 0.000000
SEQuence:ELEM2:LENGth 16384
SEQuence:ELEM2:LOOP:COUNt INFinite
SEQuence:ELEM2:WAITEvent NONE
```

SEQuence:ELEM2:JUMPEvent TIMer
SEQuence:ELEM2:JUMPTOMode NEXT
SEQuence:ELEM2:JUMPTOEntry 3
SEQuence:ELEM2:GOTOMode NEXT
SEQuence:ELEM2:GOTOEntry 1
SEQuence:ELEM2:PATTERNJUMPTOMode NEXT
SEQuence:ELEM2:PATTERN 1
SEQuence:ELEM2:PATTERNJUMPTOEntry 1
DISPlay:UNIT:VOLT AMPLitudeoff
SEQuence:ELEM3:WAVeform1 "SINC"
SEQuence:ELEM3:AMPlitude1 2.000000
SEQuence:ELEM3:OFFset1 0.000000
SEQuence:ELEM3:WAVeform2 "SINC"
SEQuence:ELEM3:AMPlitude2 2.000000
SEQuence:ELEM3:OFFset2 0.000000
SEQuence:ELEM3:LENGth 16384
SEQuence:ELEM3:LOOP:COUNt INFinite
SEQuence:ELEM3:WAITEvent MANual
SEQuence:ELEM3:JUMPEvent NONE
SEQuence:ELEM3:JUMPTOMode PREVious
SEQuence:ELEM3:JUMPTOEntry 1
SEQuence:ELEM3:GOTOMode NEXT
SEQuence:ELEM3:GOTOEntry 1
SEQuence:ELEM3:PATTERNJUMPTOMode NEXT
SEQuence:ELEM3:PATTERN 123
SEQuence:ELEM3:PATTERNJUMPTOEntry 1
TRIGger:SOURce TIMER
TRIGger:LEVel 1.000000
TRIGger:SLOPe NEGative
TRIGger:IMPedance 50Ohm
TRIGger:TIMer 4.000000
OUTPut1:STATe ON
OUTPut2:STATe ON
AWGControl:RUN
*TRG
AWGControl:DJStrobe 123