VXI Technology VM3608A

8-Channel, 16-Bit DAC/Waveform Generator

In Stock

Used and in Excellent Condition

Open Web Page

https://www.artisantg.com/47227-33

ARTISAN® TECHNOLOGY GROUP

Your **definitive** source for quality pre-owned equipment.

Artisan Technology Group

(217) 352-9330 | sales@artisantg.com | artisantg.com

- Critical and expedited services
- In stock / Ready-to-ship
- We buy your excess, underutilized, and idle equipment
- Full-service, independent repair center

VM3608A/ VM3616A

DIGITAL-TO-ANALOG CONVERTER

USER'S MANUAL

P/N: 82-0034-000
Released June 9th, 2016

VTI Instruments Corp.

2031 Main Street
Irvine, CA 92614-6509
(949) 955-1894

TABLE OF CONTENTS

CERTIFICATION

VXI Technology, Inc. (VTI) certifies that this product met its published specifications at the time of shipment from the factory. VTI further certifies that its calibration measurements are traceable to the United States National Institute of Standards and Technology (formerly National Bureau of Standards), to the extent allowed by that organization's calibration facility, and to the calibration facilities of other International Standards Organization members.

WARRANTY

The product referred to herein is warranted against defects in material and workmanship for a period of three years from the receipt date of the product at customer's facility. The sole and exclusive remedy for breach of any warranty concerning these goods shall be repair or replacement of defective parts, or a refund of the purchase price, to be determined at the option of VTI.

For warranty service or repair, this product must be returned to a VTI authorized service center. The product shall be shipped prepaid to VTI and VTI shall prepay all returns of the product to the buyer. However, the buyer shall pay all shipping charges, duties, and taxes for products returned to VTI from another country.

VTI warrants that its software and firmware designated by VTI for use with a product will execute its programming when properly installed on that product. VTI does not however warrant that the operation of the product, or software, or firmware will be uninterrupted or error free.

LIMITATION OF WARRANTY

The warranty shall not apply to defects resulting from improper or inadequate maintenance by the buyer, buyer-supplied products or interfacing, unauthorized modification or misuse, operation outside the environmental specifications for the product, or improper site preparation or maintenance.

VTI shall not be liable for injury to property other than the goods themselves. Other than the limited warranty stated above, VTI makes no other warranties, express, or implied, with respect to the quality of product beyond the description of the goods on the face of the contract. VTI specifically disclaims the implied warranties of merchantability and fitness for a particular purpose.

RESTRICTED RIGHTS LEGEND

Use, duplication, or disclosure by the Government is subject to restrictions as set forth in subdivision (b)(3)(ii) of the Rights in Technical Data and Computer Software clause in DFARS 252.227-7013.

VXI Technology, Inc.
2031 Main Street
Irvine, CA 92614-6509 U.S.A.

DECLARATION OF CONFORMITY

Declaration of Conformity According to EN ISO/IEC 17050-1:2004

MANUFACTURER'S NAME	VTI Instruments
MANUFACTURER'S ADDRESS	2031 Main Street Irvine, California 92614-6509
PRODUCT NAME	Digital-to-Analog Converter
MODEL NUMBER(S)	VM3608A & VM3616A
PRODUCT OPTIONS	All
PRODUCT CONFIGURATIONS	All

VTI Instruments (formerly VXI Technology) declares that the aforementioned product conforms to the requirements of the Low Voltage directive (European Council directive 2014/35/EU, dated 22 July 1993) and the Electromagnetic Compatibility directive (European Council directive 2014/30/EU; generally referred to as the EMC directive). In substantiation, the products were tested and/or evaluated to the standards shown below:

SAFETY	EN61010-1:2010
EMC	EN61326-1:2013 EN55011 Class A Group 1 EN61000-4-2 EN61000-4-3 EN61000-4-4 EN61000-4-5 EN61000-4-6 EN61000-4-8 EN61000-4-11 CISPR 22

June 2016

[illegible]

Steve Mauga, QA Manager

GENERAL SAFETY INSTRUCTIONS

Review the following safety precautions to avoid bodily injury and/or damage to the product. These precautions must be observed during all phases of operation or service of this product. Failure to comply with these precautions, or with specific warnings elsewhere in this manual, violates safety standards of design, manufacture, and intended use of the product.

Service should only be performed by qualified personnel.

TERMS AND SYMBOLS

These terms may appear in this manual:

WARNING	Indicates that a procedure or condition may cause bodily injury or death.
CAUTION	Indicates that a procedure or condition could possibly cause damage to equipment or loss of data.

These symbols may appear on the product:

⚠	**ATTENTION** - Important safety instructions
⏚	Frame or chassis ground
	Indicates that the product was manufactured after August 13, 2005. This mark is placed in accordance with *EN 50419, Marking of electrical and electronic equipment in accordance with Article 11(2) of Directive 2002/96/EC (WEEE)*. End-of-life product can be returned to VTI by obtaining an RMA number. Fees for take-back and recycling will apply if not prohibited by national law.

WARNINGS

Follow these precautions to avoid injury or damage to the product:

Use Proper Power Cord	To avoid hazard, only use the power cord specified for this product.
Use Proper Power Source	To avoid electrical overload, electric shock, or fire hazard, do not use a power source that applies other than the specified voltage.
Use Proper Fuse	To avoid fire hazard, only use the type and rating fuse specified for this product.

WARNINGS (CONT.)

Avoid Electric Shock	To avoid electric shock or fire hazard, do not operate this product with the covers removed. Do not connect or disconnect any cable, probes, test leads, etc. while they are connected to a voltage source. Remove all power and unplug unit before performing any service. ***Service should only be performed by qualified personnel.***
Ground the Product	This product is grounded through the grounding conductor of the power cord. To avoid electric shock, the grounding conductor must be connected to earth ground.
Operating Conditions	To avoid injury, electric shock or fire hazard: - Do not operate in wet or damp conditions. - Do not operate in an explosive atmosphere. - Operate or store only in specified temperature range. - Provide proper clearance for product ventilation to prevent overheating. - DO NOT operate if any damage to this product is suspected. ***Product should be inspected or serviced only by qualified personnel.***

SUPPORT RESOURCES

Support resources for this product are available on the Internet and at VTI Instruments customer support centers.

VTI Instruments Corp.
World Headquarters

VTI Instruments Corp.
2031 Main Street
Irvine, CA 92614-6509

Phone: (949) 955-1894
Fax: (949) 955-3041

VTI Instruments
Cleveland Instrument Division

5425 Warner Road
Suite 13
Valley View, OH 44125

Phone: (216) 447-8950
Fax: (216) 447-8951

AMETEK Instruments Pvt. Ltd. India

4th Floor, Block A,
Divyashree NR Enclave,
EPIP Industrial Area,
Whitefield,
Bangalore – 560066 INDIA

Phone: +91 80 6782 3200
Fax: +91 80 6782 3232

Technical Support

Phone: (949) 955-1894
Fax: (949) 955-3041
E-mail: support@vtiinstruments.com

Visit http://www.vtiinstruments.com for worldwide support sites and service plan information.

SECTION 1

INTRODUCTION

INTRODUCTION

The VM3608A/3616A provides 8 or 16 independent channels of a digital to analog converter (DAC) with 16 bits of resolution. Each channel consists of a DAC combined with an output amplifier. This module is part of the VMIP™ (*VXI Modular Instrumentation Platform*) family of instruments and can be combined with up to two other modules to form a high-density VXIbus instrument that fully utilizes the capabilities of the VMIP. The instrument uses the message-based word serial interface for programming and data movement, as well as supporting direct register access for high-speed data throughput. The VM3608A/3616A command-set conforms to the SCPI standard for consistency and ease of programming.

The VM3608A/3616A is a member of the VTI VMIP family and is available as an 8 or 16-, 16 or 32- or 24 or 48-channel, singlewide VXIbus instrument. In addition to these three standard configurations, the VM3608A/3616A may be combined with any of the other members of the VMIP family to form a customized and highly integrated instrument (see Figure 1-1). This allows the user to reduce system size and cost by combining the VM3608A/3616A with two other instrument functions in a singlewide C-size VXIbus module. Figure 1-2 shows the 24- or 48-channel version of the VM3608A/3616A. The 16- or 32-channel version would not have J200 and its associated LEDs and nomenclature, while the 8- or 16-channel version would also eliminate J202.

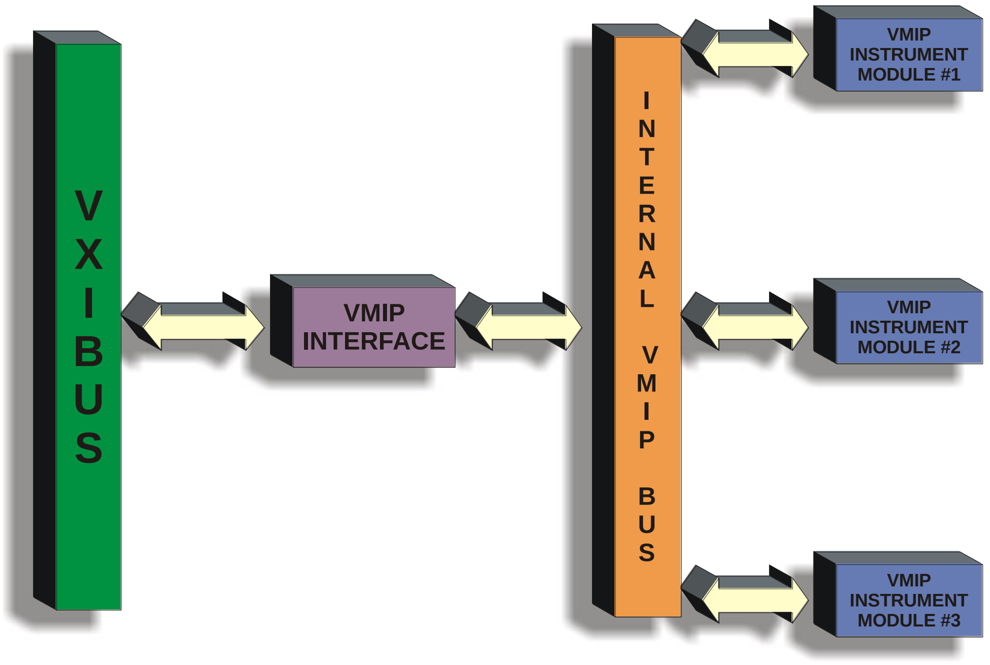
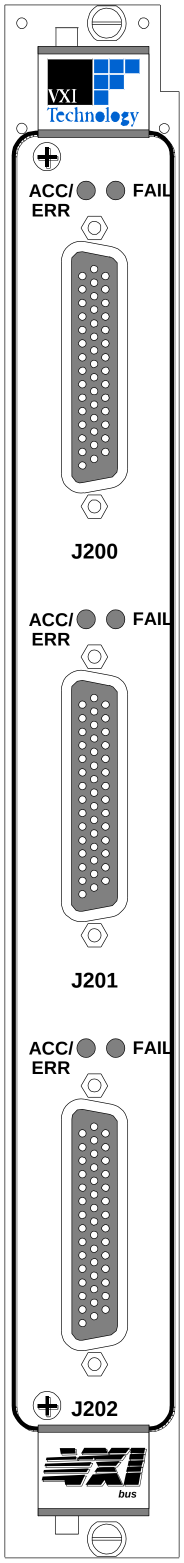

VXIBUS
VMIP INTERFACE
INTERNAL VMIP BUS
VMIP INSTRUMENT MODULE #1
VMIP INSTRUMENT MODULE #2
VMIP INSTRUMENT MODULE #3

FIGURE 1-1: VMIP™ PLATFORM



Regardless of whether the VM3608A/3616A is configured with other VM3608A/3616A modules or with other VMIP modules, each group of 16 channels is treated as an independent instrument in the VXIbus chassis and, as such, each group has its own FAIL and ACCESS light.

DESCRIPTION

The VM3608A/3616A instrumentation module provides 8 or 16 independent channels of a digital to analog converter (DAC) with 16 bits of resolution. Each channel consists of a DAC combined with an output amplifier. This module is part of the VMIP family of instruments and can be combined with up to two other modules to form a high-density VXIbus instrument that fully utilizes the capabilities of the VMIP.

In order to support accelerated testing, the module supports up to 512 predefined setups. Each setup is numbered and defines the voltage of each DAC on the board. All the voltages can then be set with a single command.

To further speed up testing, the DAC module supports up to 16 different scan lists, each 512 entries long. The scan list is a predefined sequence of setups that is loaded into the DACs. The scan list, when enabled, can be incremented through any of the trigger sources outlined in the following section.

Both the setups and scan lists are loaded into RAM prior to their use.

All the DACs update synchronously and may be triggered to update via one of three sources:

1. Trigger source from the front panel input. This input is TTL compatible and is edge sensitive. The unit may be programmed to trigger on either the rising or the falling edge of this signal.

2. Trigger source from the VXI TTL trigger bus. Any one of the eight TTL trigger bus lines may be selected as the trigger source. The unit may be programmed to trigger on either the rising or the falling edge of this signal.

3. Trigger upon receipt of a word serial command. When this mode is selected, the DACs will convert when a word serial command is received by the instrument.

FIGURE 1-2: FRONT PANEL LAYOUT

DATA MODES

Along with static output operation, the VM3608A/3616A provides a FIFO mode where the selected channels can accept and output a continuous stream of data. The VM3608A/3616A also offers Arbitrary Waveform Generation (ARB) mode that supports sophisticated looping and branching to build complex waveforms without the system controller's intervention. The data may be paced out of the instrument by using either a user-supplied clock or the internal programmable timer.

CALIBRATION

The calibration constants used to correct the data values are stored in non-volatile memory. These constants are determined when the instrument is calibrated and can be changed as necessary (such as during routine calibration cycles or when the user selects a new gain setting and wishes to set the gain accurately). These constants may also be queried at any time via a word serial query and altered via a word serial command. All calibration is done using calibration DACs to adjust the gain and offset of each channel. This eliminates the need for removing covers from the unit and allows for automated calibration.

VM3608A/3616A Specifications

General Specifications	
Number of Channels	
VM3608A/3616A-1	16 channels
VM3608A/3616A-2	32 channels
VM3608A/3616A-3	48 channels
Output Ranges	
	±20.0 V, 610 µV step, or ±10.0 V, 305 µV step
Output Current	
	±50 mA per channel (1.5 A max., total for all channels per C-size VXIbus card)
Resolution	
	16 bits, 16 bits monotonic
Short circuit	
	Continuous duration
Slew Rate	
	6 V/µs (50 mA load)
Settling Time	
	10 µs to 0.1% of specified value
Voltage Accuracy*	
10 V Range	3.22 mV
20 V Range	6.44 mV
Conversion Rate	
	100,000 voltage changes/second, normal mode register access, FIFO mode, and ARB mode 100 voltage changes per second, normal mode word serial access 2,000 voltage changes per second, single VM3616A, scan mode 1,000 voltage changes per second, two VM3616As, scan mode 667 voltage changes per second, three VM3616As, scan mode
Static Mode Update Sources	
	VXIbus TTL trigger bus 0 - 7 Front panel input, TTL compatible Updating any selected channel Word serial command
Memory	
	512 k words, 1 M word optional
FIFO Memory	
	512 k words divided by the number of active channels, 1 M word optional
FIFO Clock Source	
	Any trigger source
ARB (Arbitrary Wave Form) Memory	
	484 k words divided by the number of active channels, 996 k words optional
ARB Data Traces	
	1 to 4096 unique patterns
ARB Segments	
	1 to 4096
ARB Loop Counter	
	1 to 1048575 or Continuous
ARB Advance Modes	
	Synchronous, waits for the end of the current pattern to advance to the next Asynchronous, advances immediately to the next pattern upon being triggered

VM3608A/3616A SPECIFICATIONS (CONTINUED)	
ARB ADVANCE CONDITIONS	
	Automatic or Triggered
ARB MARKER FUNCTION	
	Marks the first data in a pattern when enabled. Polarity is software programmable.
ARB MARKER OUTPUT	
	VXIbus TTL trigger bus 0 - 7 Front panel TTL compatible output
TRIGGER SOURCE	
	VXIbus TTL trigger bus 0 - 7 Front panel input, TTL compatible Internal Timer Word serial command
ADVANCE CLOCK SOURCE	
	VXIbus TTL trigger bus 0 - 7 Front panel input, TTL compatible Internal Timer Word serial command
POWER REQUIREMENTS	
VM3608A/3616A-1	+5 V @ 1.04 A, -5.2 V @ 0.15 A, +24 V @ 0.27 A, -24 V @ 0.23 A
VM3608A/3616A-2	+5 V @ 1.34 A, -5.2 V @ 0.25 A, +24 V @ 0.54 A, -24 V @ 0.46 A
VM3608A/3616A-3	+5 V @ 1.64 A, -5.2 V @ 0.35 A, +24 V @ 0.81 A, -24 V @ 0.69 A

**** Note: Indicates that values were attained after a 30 minute warm-up.***

SECTION 2

PREPARATION FOR USE

INSTALLATION

When the VM3608A/3616A is unpacked from its shipping carton, the contents should include the following items:

(1) VM3608A/3616A VXIbus module.
(1) VM3608A/3616A Digital to Analog Converter Module User's Manual (this manual).

All components should be immediately inspected for damage upon receipt of the unit.

Once the VM3608A/3616A is assessed to be in good condition, it may be installed into an appropriate C-size or D-size VXIbus chassis in any slot other than slot 0. The chassis should be checked to ensure that it is capable of providing adequate power and cooling for the VM3608A/3616A. Once the chassis is found adequate, the VM3608A/3616A's logical address and the backplane jumpers of the chassis should be configured prior to the VM3608A/3616A's installation.

CALCULATING SYSTEM POWER AND COOLING REQUIREMENTS

It is imperative that the chassis provide adequate power and cooling for this module. Referring to the chassis user's manual, confirm that the power budget for the system (the chassis and all modules installed therein) is not exceeded and that the cooling system can provide adequate airflow at the specified backpressure.

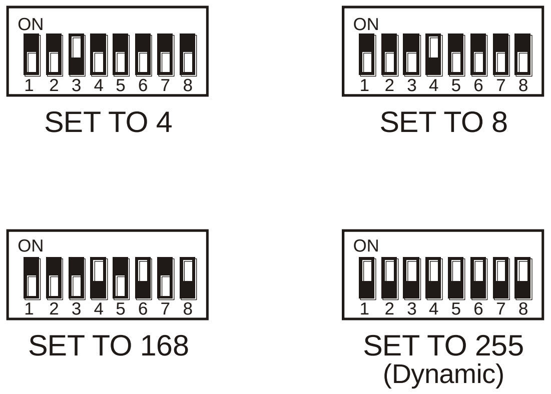
It should be noted that if the chassis cannot provide adequate power to the module, the instrument may not perform to specification or possibly not operate at all. In addition, if adequate cooling is not provided, the reliability of the instrument will be jeopardized and permanent damage may occur. Damage found to have occurred due to inadequate cooling would also void the warranty of the module.

SETTING THE CHASSIS BACKPLANE JUMPERS

Please refer to the chassis user's manual for further details on setting the backplane jumpers.

Setting the Logical Address

The logical address of the VM3608A/3616A is set by a single 8-position DIP switch located near the module's backplane connectors (this is the only switch on the module). The switch is labeled with positions 1 through 8 and with an ON position. A switch pushed toward the ON legend will signify a logic 1; switches pushed away from the ON legend will signify a logic 0. The switch located at position 1 is the least significant bit while the switch located at position 8 is the most significant bit. See Figure 2-1 for examples of setting the logical address switch.



Switch Position	Switch Value
1	1
2	2
3	4
4	8
5	16
6	32
7	64
8	128

Figure 2-1: Logical Address Switch Setting Examples

The VMIP may contain three separate instruments and will allocate logical addresses as required by the VXIbus specification (revisions 1.3 and 1.4). The logical address of the instrument is set on the VMIP carrier. The VMIP logical addresses must be set to an even multiple of 4 *unless dynamic addressing is used*. Switch positions 1 and 2 must always be set to the OFF position. Therefore, only addresses of 4, 8, 12, 16, ... 252 are allowed. The address switch should be set for one of these legal addresses and the address for the second instrument (the instrument in the center position) will automatically be set to the switch set address plus one; while the third instrument (the instrument in the lowest position) will automatically be set to the switch set address plus two. If dynamic address configuration is desired, the address switch should be set for a value of 255 (All switches set to ON). Upon power-up, the slot 0 resource manager will assign the first available logical addresses to each instrument in the VMIP module.

If dynamic address configuration is desired, the address switch should be set for a value of 255. (All switches set to ON). Upon power-up, the slot 0 resource manager will assign the first available logical addresses to each instrument in the VMIP module.

An application note is provided at the end of this manual to aid in using secondary addressing if a GPIB/Slot 0 is being used.

Front Panel Interface Wiring

The 8/16-channel version (VM3608A/3616A-1) will have J201 that contains all signals for this instrument. The 16/32-channel version (VM3608A/3616A-2) will have J201 and J202 provided, while the 24/48-channel version (VM3608A/3616A-3) will have J200, J201 and J202. The wiring for each of these connectors is identical and since each group of 8/16 channels is treated as a separate instrument, the module will have three Channel 1s, three Channel 2s, three Channel 3s, etc.

TABLE 2-1: PIN ASSIGNMENTS FOR MODELS VM3608A AND VM3616A DAC

SIGNAL	PIN	SIGNAL	PIN	SIGNAL	PIN	SIGNAL	PIN
Output1	1	Gnd	12	Gnd	23	Output6	34
Gnd	2	Marker out	13	Output11	24	Gnd	35
Gnd	3	Gnd	14	Gnd	25	Output9	36
Output4	4	Trigin*	15	Output14	26	Gnd	37
Gnd	5	Output2	16	Gnd	27	Gnd	38
Output7	6	Gnd	17	Gnd	28	Output12	39
Gnd	7	Gnd	18	Output16	29	Gnd	40
Gnd	8	Output5	19	Gnd	30	Output15	41
Output10	9	Gnd	20	Output3	31	Gnd	42
Gnd	10	Output8	21	Gnd	32	Ext Clock In	43
Output13	11	Gnd	22	Gnd	33	Gnd	44

The connector for the VM3608A and VM3616A DAC boards is a 44-pin female high-density D-sub type. Connections listed are for the VM3616A 16-channel DAC board. Outputs 9 through 16 are not used on the Model VM3608A 8-channel DAC board. A solder pot type mating connector is provided with each unit.

The pin locations for J200, J201, and J202 are shown in Figure 2-2.

WARNING **The VM3608A/3616A add a *Marker Output* and *External Clock* Input connection to the front panel connector, which were wired to ground in the legacy 3608/3616 DACs. In applications where the VM3608A/3616A replace legacy VM3608/3616 DACs, system wiring must be reviewed to prevent loading the Marker Output. Although the External Clock Input will not be damaged if connected to ground, the pin is not internally wired to ground and will not carry ground current.**

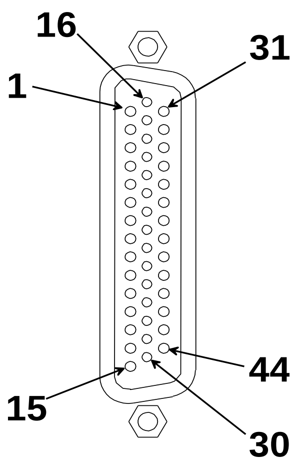


FIGURE 2-2: J200, J201 AND J202 PIN LOCATIONS

Mating Connectors

The mating connector for the VM3608A/3616A is available from the following companies:

ITT Cannon	**(www.ittcannon.com)**
P/N ZDBA44P	Connector
P/N ZD110238-1009	Connector Pin
P/N 995-2000-022	Crimp Tool
P/N 980-0004-804	Insertion/Extraction Tool

Positronic Industries, Inc.	**(www.positronic.com)**
P/N ODD44M1FY0C	Connector
P/N 9507	Crimp Tool
P/N 9502-4	Contact Positioner
P/N M81969/1-04	Insertion/Extraction Tool

SECTION 3

PROGRAMMING

INTRODUCTION

The VM3608A/3616A is a VXIbus message-based device whose command set is compliant with the Standard Command for Programmable Instruments (SCPI) programming language.

All module commands are sent over the VXIbus backplane to the module. Commands may be in upper, lower, or mixed case. All numbers are sent in ASCII decimal unless otherwise noted.

The module recognizes SCPI commands. SCPI is a tree-structured language based on IEEE-STD-488.2 Specifications. It utilizes the IEEE-STD-488.2 Standard command, and the device dependent commands are structured to allow multiple branches off the same trunk to be used without repeating the trunk. To use this facility, terminate each branch with a semicolon. As an example, **SLOPe**, and **SOURce** are both branches off the **TRIGger:** trunk and can be combined as follows:

```
TRIGger:SLOPe POS;SOURce EXT
```

The above command is the same as these two commands:

```
TRIGger:SLOPe POS
TRIGger:SOURce EXT
```

See the Standard Command for Programmable Instruments (SCPI) Manual, Volume 1: Syntax & Style, Section 6, for more information.

The SCPI commands in this section are listed in upper and lower case. Character case is used to indicate different forms of the same command. Keywords can have both a short form and a long form (some commands only have one form). The short form uses just the keyword characters in uppercase. The long form uses the keyword characters in uppercase plus the keyword characters in lowercase. Either form is acceptable. Note that there are no intermediate forms. All characters of the short form or all characters of the long form must be used. Short forms and long forms may be freely intermixed. The actual commands sent can be in upper case, lower case or mixed case (case is only used to distinguish short and long form for the user). As an example, these commands are all correct and all have the same effect:

```
TRIGger:SOURce EXTernal
trigger:source external
TRIGGER:SOURCE EXTERNAL
TRIG:SOURce EXTernal
TRIG:SOUR EXTernal
TRIG:SOUR EXT
trig:sour EXT
trig:sour ext
```

The following command is **not** correct because it uses part of the long form of **TRIGger**, but not all the characters of the long form:

TRIG~~G~~:SOUR EXT (incorrect syntax - extra "g"- only trig or trigger is correct)

All of the SCPI commands also have a query form unless otherwise noted. Query forms contain a question mark (?). The query form allows the system to ask what the current setting of a parameter is. The query form of the command generally replaces the parameter with a question mark (?). Query responses do not include the command header. This means only the parameter is returned: no part of the command or "question" is returned.

When character data is used for a parameter, both short and long forms are recognized. If the command has a query form with character response data, the short form is always returned in upper case. As an example, to find out what the current trigger source setting is use the following command:

```
TRIG:SOUR?
```

The response would be:

```
EXT
```

This tells the user that the trigger is set to an external source.

Multiple commands can also be combined on one line. To do this, terminate one command with a semicolon and start the next command with a colon. As an example, the trigger source can be set to a positive edge and an output trigger line can be enabled as follows:

```
TRIG:SOUR EXT;:OUTPUT:TTLTRG1 ON
```

The IEEE-STD-488.2 Common Commands can be placed anywhere set off from the rest of the command by a semicolon. They can also be placed alone on a line. For example, place the ***rst** command in front of a setting string as follows:

```
*RST;OUT:TTLT 3
```

Note that the **OUT:TTLT** command set did not require a leading colon (:) because there was no prior trunk of the SCPI tree.

NOTATION

Keywords or parameters enclosed in square brackets ([ ]) are optional. If the optional part is a keyword, the keyword can be included or left out. Omitting an optional parameter will cause its default to be used.

Parameters are enclosed by angle brackets (< >). Braces ({ }),or curly brackets, are used to enclose one or more parameters that may be included zero or more times. A vertical bar (|), read as "or", is used to separate parameter alternatives.

DATA SETUP AND SCAN LISTS

Normal Mode

In normal mode, all channels are set to static values via word serial command or through direct register access. To support higher throughput, 512 predefined values for each channel may be loaded into RAM. With a single-word serial command, all 8 or 16 channels are updated at once. To facilitate synchronization, each channel may be set to change value when written to, or only when one or more selected channels are written to. This allows multiple values to be changed, but all channels will update at the same time. Updates can also be synchronized to the front panel trigger input, the VXIbus trigger lines, or via a word serial command.

FIFO Mode

In FIFO mode, any or all channels may be set up to output a continuous stream of data, while all other channels remain at their previously programmed values. Data is fed into the instrument using direct register access and data throughput of up to 100k samples per second. The output may be paced using an internal timer with 100 ns resolution, the VXIbus trigger lines, or via a front panel trigger input.

Arbitrary Waveform Generator Mode

In Arbitrary Waveform Generator (ARB) mode, any or all channels may be set up to output complex waveforms by loading one or more waveform segments and linking them together. All other channels remain at their previously programmed values. Up to 4096 waveform segments may be loaded into the instrument and 4096 segments may be linked together. Each segment may be used once, looped up to over a million times, or may loop continuously until triggered to advance to the next segment. Advancing from segment to segment can occur automatically, or when a trigger event is received. In ARB mode, all enabled channels track each other and all waveform segments for each channel are the same length. All lopping and branching instructions apply for all enabled channels. The data format should be set prior to configuring the scan mode for ARB operation.

Scan Mode

In scan mode, any or all channels may be set up to output data previously stored in RAM. Each channel may have up to 512 data points, and each channel may have a unique number of points. The data is advanced by a VXIbus TTL trigger event, the front panel trigger input, or by word serial command. Scan mode may be set up to either stop at the end of the data set, or restart from the beginning of the data set. In this mode, the output levels of the channels that are not included in scan mode may be altered while scan mode is running.

EXAMPLES OF SCPI COMMANDS

CALibration:COUNt?

The Calibration Count query returns a number that indicates the number of times the VM3608A/3616A has been calibrated. The instrument will increment the count every time the non-volatile memory storing the calibration constants is updated. The non-volatile memory has a guaranteed minimum of 10,000 cycles.

CALibration:COUNt?	**Where the maximum value for count is 16,777,215 after which it will wrap to 0. There are no query parameters.**

EXAMPLES

`CAL:COUN?`	`3` (Returns a number 3, which indicates the VM3608A/ 3616A has been calibrated 3 times.)
CAL:SEC:STAT off,#17VM3616A	Disabling Security
`CAL:GAIN 3`	Programming Channel 1's gain
`CAL1:ZERO 2`	Programming Channel 1's zero
`CAL:STOR`	Storing calibration constants to non-volatile memory
`CAL:SEC:STAT ON`	Enabling security
`CAL:COUN?`	`4` (Querying calibration count after storing the new configuration)

CALibration:DATA

The Calibration Data command is used to send the calibration constants to the VM3608A/3616A in indefinite or definite length arbitrary block format. The whole data set must be provided, otherwise an error will be generated. It is important to note that the calibration constants will change only if the calibration security is off.

CALibration:DATA< block_data>	**Where <block_data> is a block of data in IEEE-488.2 definite or indefinite length arbitrary block format. The calibration constants are implemented as 8-bit values from -127 to +128.**

EXAMPLES

CAL:SEC:STAT OFF,#17VM3616A	*Disabling Security*
CAL:DATA #232 <block_data>	*Sending the data block*
	Where <block_data> in decimal format would look like: 12300174011021230014367192100156
CAL:SEC:STAT ON	Enabling Security

CALibration:GAIN

The Calibration Gain command is used to set the calibration constant for the gain of the selected channel and its effect is immediate. The calibration gain command will function only when the calibration security is disabled, otherwise an error is generated.

CALibration<channel>:GAIN<value>	**Where <channel> is 1 through 16 or 1 through 8, referring to a specific calibration DAC.**
	Where <value> is -128 to +127.

EXAMPLES

`CAL2:GAIN 50`	Sets the gain value for Channel 2 to 50.
`CAL2:GAIN?`	50 (Returns the gain value for Channel 2, which is currently set to 50.)

CALibration:SECure:CODE

The Calibration Secure Code command sets the code required to disable the calibration security. The calibration security must be disabled in order to change the code string. The default code set by the factory for VM3616A is 'VM3616A' and for VM3608A is 'VM3608A'.

CALibration:SECure:CODE<code>	**Where <code> can be from 1 to 12 ASCII characters in length entered in IEEE-488.2 definite or indefinite length arbitrary block format.**

EXAMPLES

CALibration:SECure:CODE #17VM3616	Sets the security code for VM3616A in IEEE-488.2 definite or indefinite length arbitrary block format.
CALibration:SECure:CODE?	#16VM3616 (Returns the calibration security code set as "VM3616".

CALibration:SECure:STATe

The Calibration Secure State command enables or disables the calibration security. When security state is on or active, the calibration constants may not be stored to the non-volatile memory. To store the calibration constants to the non-volatile memory, security state must be off or disabled. In order to disable security state, the code must be supplied and must be in four-part block format. The four parts are:

1) #
2) A single digit that tells how many digits are in the length.
3) The length of the security code.
4) The actual data (in this case, the character of the password).

CALibration:SECure:STATe<mode>,<code>	**Where <mode> is 0 | 1 | OFF | ON. 0 or OFF means values may be stored in the non-volatile memory. 1 or ON means values may not be stored in the non-volatile memory.** **Where <code> is the parameter that must be present to disable the security which comprises of four parts as described above.**

EXAMPLES

CALibration:SECure:STATe OFF,#17VM3616A	Sets the security state off so that the calibration constants can be stored in the non-volatile memory. Note that the password is case sensitive.
CALibration:SECure:STAT?	0 (Reports the state of the security, which is currently set as OFF.)
CALibration:SECure:STATe ON	Sets the security state ON so that calibration constants cannot be stored in the non-volatile memory.
CALibration:SECure:STATe?	1 (Reports the state of the security, which is currently set as ON.)

CALibration:STORe

The Calibration Store command saves the current calibration constants into the non-volatile memory when the CAL:SEC:STAT is OFF. This command has no effect on the non-volatile memory when the CAL:SEC:STAT is ON and it generates an error.

CALibration:STORe	**The security state is to be off prior to the usage of the command.**

EXAMPLES

CALibration:SECure:STATe OFF,#17VM3640A	Sets the security state off so that the calibration constants can be stored in the non-volatile memory. Note that the password is case sensitive.
CALibration:STORe	Saves the current calibration constants to the non-volatile memory. [Assume that the security is OFF].

CALibration:STORe:AUTO

The Calibration Store auto command allows the new calibration constants, to be saved into the non-volatile memory, automatically. This command has no effect on the non-volatile memory when the CAL:SEC:STAT is ON and it generates an error.

CALibration:STORe:AUTO <mode>	**Where <mode> is 0 | OFF | 1 | ON. The command will not execute if calibration security is enabled and, an error is generated.**

EXAMPLES

CALibration:STORe:AUTO ON	Saves the new calibration constants to the non-volatile memory automatically.
CALibration:STORe:AUTO?	1 (Reports 1, stating that auto store is enabled or ON.)

CALibration:ZERO

The Calibration Zero command is used to set the calibration constant for the offset of the selected channel; its effect is immediate. The calibration zero command will function only when the calibration security is disabled, otherwise an error is generated.

CALibration<channel>:ZERO <value>	**Where <channel> is 1 through 16 or 1 through 8 referring to a specific calibration DAC.** **Where <value> is -128 to +127.**

EXAMPLES

CALibration 2:ZERO -100	Sets the zero value or offset for Channel 2 to -100.
CALibration 2:ZERO?	-100 (Reports the offset value for Channel 2 that is currently set as -100.)

FORMat

The Format command is used to set the output format for the digital queries.

FORMat<type>	**Where <type> is ASCII, HEXadecimal, OCTal, BINary.**

EXAMPLES

FORMat ASC	Sets the data output to be in ASCII format.
FORMat?	ASC (Returns the output data in ASCII format.)
FORMat BIN	Sets the data output to be in BINary.
FORMat?	BIN (Returns the output data as BINary format.)
FORMat HEX	Sets the data output to be in HEXadecimal.
FORMat?	HEX (Returns the output data as HEXadecimal format.)
FORMat OCT	Sets the data output to be in OCTal.
FORMat?	OCT (Returns the output data as OCTal format.)

MEMory:SETup

The Memory Setup command allows the entry of separate voltages that will be loaded to the Precision DACS. Each DAC channel has an associated 512-element "memory" array. The same elements in all 16 arrays are loaded at the same time from the 16-element voltage list.

MEMory:SETup<index>,<voltage_list>	**Where <index> is an integer number from 1 to 512 (which specifies the memory array element).**
	Where <voltage_list> is a list of 8 or 16 voltages.

EXAMPLES

MEMory:SETup 1,2,3,4,5	Sets the voltages at index 1 in the "memory" array
MEMory:SETup? 1	2.000122, 2.999878, 4.000244, 5.00000 (Returns the configured voltages at index 1.)

SCAN

The Scan command is used to enable or disable the operation of the scan list operation for the specified channels. For each channel that is enabled, the interrupt routine will load a voltage from its respective scan list arrays at the current array position to the DAC and auto increment the scan list array pointer. If the array pointer equals the limit, then the scan function for that channel will stop, unless the mode of that channel is set to LOOP. LOOP mode means the scan function will reset the array pointer to 0 and continue.

When the scan mode is to be enabled, a trigger source of either EXTernal or one of the TTLTriggers is required to be selected first. The TRIGer:IMMediate:ADVance must be set when using ARBitrary mode.

Note that a scan mode of a channel must first be set to OFF before another mode may be set. The SCAN OFF command would reset all channel scan modes to OFF.

SCAN<mode>,<channel_list>	**Where <mode> can be 0 | OFF | 1 | ON | LOOP | FIFO | ARB. LOOP mode means the scan function will reset the array pointer to 0 and continue.** **Where <channel_list> is a list of 8 or 16 DAC channels.**

EXAMPLES

SCAN ON, (@1:4)	Sets the scan mode to ON for Channels 1 through 4.
SCAN?2	ON (Returns the scan mode for Channel 2, which is currently set ON.)
SCAN ON,(@1)	Sets the scan mode to ON for Channel 1.
SCAN LOOP,(@1)	This would normally set the scan mode for Channel 1 as LOOP. However, as the transition from ON to LOOP is illegal, a system error would be generated.
SCAN OFF, (@1:4)	Sets the scan mode for Channels 1 through 4 to OFF.
SCAN ARB, (@1:4)	Sets the scan mode for Channels 1 through 4 to ARBitrary mode.

SCAN:LIMit

The Scan Limit command specifies a position in the 512 element scan list array, where the interrupt routine loading the DACs, should either stop of loop back to zero.

SCAN:LIMit<channel>,<count>	**Where <channel> is a specific DAC channel of 1 through 8 or 1 through 16.**
	Where <count> is a position in the scan list array (ranging from 1 to 512) to stop or loop back.

EXAMPLES

SCAN:LIMit 2, 256	Sets the scan limit for Channel 2 to 256.
SCAN:LIMit?2	256 (Reports the scan limit of the scan list array for Channel 2, which is currently set to 256.)

SCAN:TABLe

The Scan Table command loads the scan list of a specific channel with voltage values. These voltage values are then loaded to the DAC by the interrupt routine. This operation is dependent on scan mode and scan limit. It is important to note that the instrument does not have an on-board clock. So, in order to set up the scan mode, the trigger source that triggers through the scan list must also be specified. The valid trigger sources for the SCAN mode are TTLTO-7 and EXTernal.

SCAN:TABLe <channel>,<voltage_list>	**Where <channel> is a specific DAC channel of 1 through 16 or 1 through 8.** **Where <voltage_list> is a list of voltages to be loaded in the channel scan list. The number of voltages ranges from 1 to 512.**

EXAMPLES

SCAN:TABLe 1,2,3,4,5	Loads Channel 1 with the voltage values of 2, 3, 4 and 5.
SCAN:TABLe?1 3, 2	2.999878,4.000244,5.00000 (Returns 3 voltages from Channel 1’s scan list array starting at position 2.)

SCAN:TABLe:LOCation

The Scan Table Location command allows a voltage value at a specific location in the scan list array, to be modified or queried.

SCAN:TABLe<channel>:LOCATION <number>,<voltage>

Where <channel> is a specific DAC Channel 1 through 16 or 1 through 8.

Where <number> is a specific location in the scan list array.

Where <voltage> is a single voltage value.

EXAMPLES

SCAN:TABLe 1:LOCation 2,4	Enters a voltage value 4, at a specific location 2 in Channel 1's scan list array.
SCAN:TABLe 1:LOCation? 2	4.000244 (Returns a voltage value of 4.000244 at a specific location (2) in the scan list array of Channel 1.)

SOURce:VOLTage:DATA

The Source Voltage Data command sets the output level of the channels selected by the channel list using the specified data. Note that the 16-bit precision DAC is programmed with the binary value specified.

NOTE The following applies when the unit is in 20 V range and the SOURce:VOLTage:FORMat command mode is OFF.

SOURce:VOLTage:DATA<value>,<channel_list>	**Where <value> is the voltage value from -32768 to +32767, i.e. -20.00000 to +19.99939 in binary.** **Where <channel_list> is a list of channels to be loaded with a specified voltage value.**

EXAMPLES

SOURce:VOLTage:DATA 16384,(@1, 2, 3)	Loads the Channels 1, 2, 3 with a voltage value of 16384 (ASCII) which is equivalent to 10 V.
SOURce:VOLTage:DATA? 3	16384 (Reports the voltage value of Channel 3, which is currently set as 16384, i.e., 10 V.)

SOURce:VOLTage:LEVel

The Source Voltage Level sets the output voltage level of channels selected by the channel list.

SOURce:VOLTage:LEVel<value>,<channel_list>	**Where <value> is the voltage values ranging from -20.00000 to +19.99939**
	Where <channel_list> is a list of channels from 1 through 16 or 1 through 8, to be loaded with a specified voltage value.

EXAMPLES

SOURce:VOLTage:LEVel 2,(@1,2,3)	Loads the Channels 1, 2, 3 with the specified voltage value of 2.
SOURce:VOLTage:LEVel? 2	2.999878 (Returns the voltage value of Channel 2.)

SOURce:VOLTage:SETup

The Source Voltage Setup command loads each DAC with a voltage value from the location in its respective "memory lists," specified by <index>. The memory list is set up using the MEMory:SETup command.

SOURce:VOLTage:SETup<index>	**Where <index> is the location in the "memory" list ranging from 1 to 512.**

EXAMPLES

SOURce:VOLTage:LEVel 1	Loads the DAC with a voltage value at index 1.

TRIGger:SLOPe

The Trigger Slope command selects which edge of a triggering signal is the active edge.

TRIGger:SLOPe<slope>	**Where <slope> may be positive or negative.**

EXAMPLES

TRIGger:SLOPe POS	Selects the active edge for a triggering signal to be POSitive.
TRIGger:SLOPe?	POS (Returns the active edge for a triggering signal, which is currently set as POSitive.)
TRIG:SLOP NEG	Selects the active edge for a triggering signal to be a negative edge.
TRIG:SLOP?	NEG (Returns the active edge for a triggering signal, which is currently negative.)

TRIGger:SOURce

The Trigger Source command selects the trigger event that updates the DACs on the VM3608A/3616A. Each DAC is double buffered and hence, writing to the DAC will require a second event to cause output voltage to be updated.

TRIGger:SOURce <source>	**Where <source> can be one of the following: NONE, INTernal 1 - 16, AUTO, EXTernal and TTLTrg 0 - 7.**

EXAMPLES

TRIGger:SOURce TTLT3	Selects the trigger source to be TTLTrigger 3.

NOTE While using SCAN, a trigger source of either EXTernal or one of the TTLTriggers is required. The trigger source command is used to select the above required trigger sources.

For the VM3608A, only trigger sources INTI-8 are valid internal triggers. However for the VM3616A, INTI-16 are valid internal trigger sources.

APPLICATION EXAMPLES

This section contains examples of using SCPI command strings for programming the VM3608A/3616A module. The code is functional and will contain a brief description of the operation.

In this example, the VM3608A/3616A sets the calibration gain and calibration offset for the specified channel. The calibration security state is turned OFF, then the changes are stored to the non-volatile memory. It also returns the number of times, the non-volatile memory is updated.

CAL:SEC:STAT 0,#17VM3608A	Sets the calibration security state OFF with code VTI.
CAL 1:GAIN 1	Sets the calibration gain for Channel 1 to 1 V.
CAL 2:ZERO 1	Sets the calibration offset for Channel 2 to 1 V.
CAL:STOR	Stores the calibration gain and offset values to the non-volatile memory.
CAL:COUNT?	Returns 5, stating that the non-volatile memory has been updated 5 times.
CAL:SEC:STAT 1	Disables further stores to non-volatile memory.

In this example, the VM3608A/3616A sets up the trigger source and output voltage level for a selected list of channels.

TRIG:SOUR TTLT3	Sets the trigger source to TTLT3.
TRIG:SLOP POS	Sets the trigger slope to positive.
SOUR:VOLT:LEV -10, (@7:10)	Sets the output voltage level of -10.00000 V for the selected list of channels from 7 through 10.

In this example, the VM3608A/3616A loads scan list of the specified channel with voltage values. These values can be loaded to the DAC using the interrupt routine-Note, for the above to function, the trigger source should be either EXTernal or TTLTrigger 0 - 7.

TRIG:SOUR EXT	Sets the trigger source as EXT.
SCAN 1, (@1:4)	Sets the scan mode ON for Channels 1 through 4.
SCAN:L1M 1,256	Sets the position in the scan list arrays for Channel 1 as 256.
SCAN:TAB 1,2,3,4,5	Loads the Channel 1's scan list with voltage values 2, 3, 4 and 5.

In this example, the memory list of the VM3608A/3616A is setup and then the output levels of the channels are programmed using the memory lists.

MEM:SETUP 1, 2,3,4,5,6,7,8,10	Setting up memory list of Channels 1 through 8 at index 1 assuming VM3608A.
MEM:SETUP 2, 10,11,12,14,15,12,11,1	Setting up memory list of Channels 1 through 8 at index 2 assuming VM3608A.
SOUR: VOLT: SETUP 1	Setting the DAC output levels to the voltages programmed in memory lists at index 1.

REGISTER ACCESS

The VM3608A/3616A module provides direct register access for faster data access. The DAC output values may be programmed using direct register access providing maximum speed. In FIFO mode, the data stream is sent to the instrument via direct register access. In Arbitrary Waveform Generator (ARB) mode, the waveform data may be input using either word serial data access or register-based data access.

The register map is as shown in Table 3-1 below.

PROGRAMMING THE DACS VIA REGISTER ACCESS

In order to program a particular channel to a particular voltage value, the equivalent 16-bit representation must be computed first. Second, the computed value must be written to the register at the offset corresponding to that channel. When **SOURce:VOLTage:FORMat** is set to 1 or ON, the Register value is calculated as:

The nearest integer value of: ((Voltage * 65535) / Full Range + 32768)

Where:

Voltage is the desired output value

Full Range is the total of Maximum positive output + |Maximum Negative Output|

For the 20 V range, this value is 39.99939

For the 10 V range, this value is 19.99969

For example, if Channel 1 is set to the 20V range, and is to be programmed to 10volts, compute the 16-bit equivalent representation for 10 volts as follows:

((10 * 65535) / 39.99939) + 32768 = 49151.999856 ~ 49152

When **SOURce:VOLTage:FORMat** is set to 0 or OFF, the Register value is calculated as:

The nearest integer value of: (Voltage * 65535) / Full Range

When the decimal result is converted into a 16-bit hexadecimal value, the format will be

#H0 to #H7FFF = 0 to (<range> - LSB),

and #H8000 to #HFFFF = -<range> to (0 - LSB);

where <range> is 10V for the 10V range, and 20V for the 20V range,

*and LSB = (2 * <range>) / 65536.*

For example, if Channel 1 is set to the 20 V range, and is to be programmed to -10 V, compute the 16-bit equivalent representation for -10 V as follows:

(-10 * 65535) / 39.99939 = -16383.999856 ~ -16384

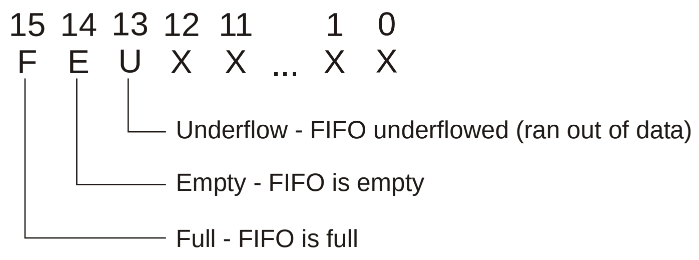
Converted to hexadecimal -16384 = #HC000.

The VM3608A/3616A DAC Module supports access to the 16/8 output channels via the Device Dependent Registers of the VXIbus interface. The following table shows A16 Memory and the VM3608A/3616A Data Port Map. Note that channels 9 through 16 are not available in the VM3608A DAC.

TABLE 3-1: A16 MEMORY

Offset	Description
3E	Channel 16
3C	Channel 15
3A	Channel 14
38	Channel 13
36	Channel 12
34	Channel 11
32	Channel 10
30	Channel 9
2E	Channel 8
2C	Channel 7
2A	Channel 6
28	Channel 5
26	Channel 4
24	Channel 3
22	Channel 2
20	Channel 1/FIFO DATA
1E	
1C	
1A	
18	
16	[A32 Pointer Low]
14	[A32 Pointer High]
12	[A24 Pointer Low]
10	[A24 Pointer High]
E	Data Low
C	Data High
A	Response [/Data Extended]
8	Protocol [/Signal] Register
6	[Offset Register]
4	Status / Control Register
2	Device Type
0	ID Register

A READ of Register 20 will return information concerning FIFO read status. The information is returned in the following format:



VXI*plug&play* DRIVER

PROGRAMMING THE VM3608A/3616A USING VXI*plug&play* DRIVERS

1) Outputting single values
2) Using the Scan Table
3) ARB mode
4) FIFO mode

OVERVIEW

The VM3608A/3616A is a very flexible instrument. As such, the VXI*plug&play* drivers supporting the instrument have been designed to give the user maximum access to all of the capabilities of the instrument. However, VXI*plug&play* drivers by their high level nature and associated overhead, do not give the programmer complete freedom or the ability to enhance the operating speed of the device. For example, the VXI*plug&play* drivers do not give the programmer access to the device registers.

The VM3608A/3616A has four different modes that determine how the output of the DAC is updated. It can act as a static analog output where the value is set dependent on the execution of a line of code. The output can also be updated where the value is dependent on an element in a scan table. The ARB mode allows the user to download arbitrary waveforms into memory and each channel can be programmed to output a waveform from the stored memory. The FIFO mode is used to continuously download data to the DAC.

Each mode offers varying degrees of complexity. A number of support tools have been provided within the VXI*plug&play* driver. One of the most useful is the help file, **Vtvm3616a.hlp**. This file contains the function tree and a very detailed description of each VXI*plug&play* driver. A soft front panel is also provided with the instrument. It is a LabWindows application and the code behind the panel is included. The code provides a framework from which other applications can be developed. It is suggested that the user spend some time looking over the soft front panel and the help file to get a better understanding of instrument operation.

A description of each of the modes is provided below.

OUTPUTTING A SINGLE VALUE

There is an application function in the VXI*plug&play* driver set that allows the user to program a single voltage level on any one of the output channels.

Vtvm3616_setupAndWriteToDAC

The parameters passed to this function are the trigger source, trigger slope, voltage level value, channel list and the number of channels in the channel list. Executing this function will program the DAC at the voltage level. This function calls two other 'core' VXI*plug&play* functions and is an example of how to group them together. If the trigger source is set to AUTO, the output will be immediately updated. If the trigger is set to anything else, that event must take place for the output to update.

OUTPUTTING FROM THE SCAN LIST

Up to 512 voltage level elements can be loaded into a scan list and the DAC can be programmed to step through the list. A trigger event will cause the output of the DAC to be updated respective of the value of the next index. Once the end of the list is reached, the DAC can either loop back to the beginning of the list and repeat its sequence, or it can stop updating once the end is reached. To program the scan list parameters, a *VXIplug&play* application function has been provided. It is important to note that the scan list is not used when the instrument is set to the ARB or FIFO mode.

Vtvm3616_SetupScanListParameters

The parameters passed to this function are the channel number, mode (off, on, or loop), index in the list where the DAC will either loop back or stop, voltage list (scan list values) and the number of entries in the scan list.

Vtvm3616_ConfigTrigParameters

The trigger parameters should also be set to indicate how the scan list index will be advanced and the output subsequently updated.

The scan list is limited in its update rate capabilities because it is interrupt driven.

OUTPUTTING AN ARBITRARY WAVEFORM

To output an arbitrary waveform, it is important to understand the concept of *traces*, *segments* and *sequences*. A 100-point sine wave, a 200-point triangle wave, a 300-point ramp function and a 300 point sine/triangle waveform are examples of traces. Up to 16 channels (8 for the VM3608A) can be programmed in the ARB mode. A segment describes what each channel is outputting at a particular point in time based on the trace data loaded for that particular channel. A segment can contain up to *n* traces, where *n* is the number of channels set to ARB mode.

Each defined segment is assigned a size, or number of points that it contains. This size is consistent for all channels. The segment size always contains at least the number of data points of its largest trace that is loaded. For example, if the following continuous outputs are desired

Channel 1 - 100 point sine wave

Channel 2 - 200 point triangle wave

Channel 3 - 300 point ramp function

the segment size will be set at 600 points. Because the output needs to be continuous, six cycles of the 100 point sine wave data, three cycles of the 200 point triangle data and two cycles of the 300 point ramp function will need to be loaded into the memory corresponding to segment 0 for each of the channels. This ensures output without gaps or interruptions. Had the segment size been set to 300 points, the triangle wave would not have been continuous. If only one cycle of the 100-point sine wave had been loaded, the output of Channel 1 would complete one cycle of the sine wave and not update the DAC for the next 500 clock ticks. It is important to remember that continuous waveform outputs require definition of the entire segment for each channel. Additional segments can be defined for the channels in the ARB mode as described above. Any subsequent segments can have different numbers of data points.

A sequence list must be defined before output is generated by the DAC. The elements of a sequence list are segments. Once a segment has been defined, it can be included in the sequence list. In the example above, only one segment was defined. The sequence list describing the waveforms out of the three channels would have consisted of only one element (segment 0). Sequence lists can contain up to 4096 segments. Furthermore, segments can be repeated within the list (not necessarily 4096 unique segments). To further illustrate, a sequence list containing more than one segment would be useful in programming one of the channels to output a sine wave followed by a triangle wave and a ramp function. This assumes that the new segments have been defined.

The *VXIplug&play* sequence to properly program the VM3608A/3616A in the ARB mode is detailed below. Refer to the help file for detailed descriptions of each function and parameter.

```
/*
 *initiate the VM3616A
 */
vtvm3616_init (ViRsrc instrDesc, ViBoolean vtvm3616_ON, ViBoolean
              vtvm3616_ON, ViSession *instrHndl);

/*
 *set the voltage range
 */
vtvm3616_setupVoltageRange (ViSession instrHndl, ViInt16 range,
              ViInt16 channelList[], ViInt16 numOfChannels);

/*
 *set trigger to NONE
 */
vtvm3616_configTriggerParams (ViSession instrHndl, ViInt16
              triggerSource, ViBoolen slope);
/*
 *set the format of the binary data to be passed to the instrument (signed or unsigned binary)
 */
vtvm3616_setupDataFormat (ViSession instrHndl, ViInt16 dataFormat)

/*
 *set mode to ARB
 */
vtvm3616_setupScanMode (ViSession instrHndl, ViInt16 mode, ViInt16
              channelList[], ViInt16 numOfChannels);
```

```
/*
 *define the segments and their length
 *This line is repeated n times, where n represents the number of segments in the pattern
 */
vtvm3616_arbDefineSegment (ViSession instrHndl, int segmentNumber,
              ViInt32 buffSize0)


/*
 *define the clock source (timer = internal oscillator)
 */
vtvm3616_setupClock (ViSession instrHndl, ViInt16 clockSource,
              ViBoolean slope)

/*
 *set the timer to desired period (if internal oscillator is the clock source)
 */
vtvm3616_setupTimer (ViSession instrHndl, double period)

/*
 *load the 16-bit binary data for each segment | channel pair
 *This command will be called n times, where n = number of channels * number of segments
 */
vtvm3616_arbLoadData (ViSession instrHndl, int segmentNumber, int
              channel, ViInt16 voltageDataBuffer [], ViInt32
              buffSize)

/*
 *define the sequence list
 *This command determines the order of the segments that will be output from each channel
 */
vtvm3616_arbDefineSequence (ViSession instrHndl, ViInt16
              sequenceList[], ViInt16 numOfSegments);

/*
 *define sequence parameters, auto advance ensures there will be no breaks between segments
 */
vtvm3616_arbSetSeqParms (ViSession instrHndl, int seqIndex, int
              seqStart, int seqAdvance, int seqRepeat, int
              seqMarker)

/*
 *output the waveform
 */
vtvm3616_commandArb (ViSession instrHndl, ViBoolean command)
```

OUTPUTTING DATA USING FIFO MODE

Data can be continuously loaded to the DAC via a FIFO. The FIFO is loaded with data. When it begins to empty, based on the clock source rate, new data will need to be supplied. There are a number of applications where the FIFO mode could be used. If a large amount of data has been collected using an A/D device and stored in a file, the VM3608A/3616A can be used to 'play' the data back at the specified frequency via the FIFO interface. For example, sound files can be played out of the device and can be heard when speakers are connected to a channel output.

The following command set is used to program the DAC in the FIFO mode.

```
/*
 *initiate the VM3616A
 */
vtvm3616_init (ViRsrc instrDesc, ViBoolean vtvm3616_ON, ViBoolean
              vtvm3616_ON, ViSession *instrHndl);

/*
 *set the voltage range
 */
vtvm3616_setupVoltageRange (ViSession instrHndl, ViInt16 range,
              ViInt16 channelList[], ViInt16 numOfChannels);

/*
 *set trigger to NONE
 */
vtvm3616_configTriggerParams (ViSession instrHndl, ViInt16
              triggerSource, ViBoolean slope);

/*
 *set mode to FIFO
 */
vtvm3616_setupScanMode (ViSession instrHndl, ViInt16 mode, ViInt16
              channelList[], ViInt16 numOfChannels);

/*
 *load the FIFO data
 */
vtvm3616_loadFIFO_Data (ViSession instrHndl, ViInt16 dataBuffer[],
              ViInt32 buffSize)

/*
 *output the waveform
 */
vtvm3616_commandDIFO (ViSession instrHndl, ViBoolean command)
```

```
/*
  *use this function to determine the status if the FIFO (full|empty|under-run)
  */
vtvm3616_GetFIFOStatus (ViSession instrHndl, ViUInt16
               *fifoStatus);
```

The FIFO mode sequence is very complex. The status is checked to see when new data should be loaded. Proper care must be practiced to ensure that the FIFO does not under-run or empty itself before more data can be loaded. Refer to the help file, and the soft front panel FIFO example, to get tips on programming in this mode.

VXI*plug&play* DRIVER EXAMPLES

```
/***************************************************************************
Function:               vtvm3616_setupAndWriteToDAC
Formal Parameters       ViSession  instrHndl,
                        - A valid session handle to the instrument.

ViInt16 triggerSource,
                        - This parameter is used to select the trigger type
                        i.e., Trigger Source.

                  Valid Range:                    Interpretation:
                  -----------                     ---------------
                  vtvm3616_TRIG_SRC_AUTO          Auto
                  vtvm3616_TRIG_SRC_EXT           External
                  vtvm3616_TRIG_SRC_TTLTRG0       TTL Trigger 0
                  vtvm3616_TRIG_SRC_TTLTRG1       TTL Trigger 1
                  vtvm3616_TRIG_SRC_TTLTRG2       TTL Trigger 2
                  vtvm3616_TRIG_SRC_TTLTRG3       TTL Trigger 3
                  vtvm3616_TRIG_SRC_TTLTRG4       TTL Trigger 4
                  vtvm3616_TRIG_SRC_TTLTRG5       TTL Trigger 5
                  vtvm3616_TRIG_SRC_TTLTRG6       TTL Trigger 6
                  vtvm3616_TRIG_SRC_TTLTRG7       TTL Trigger 7
                  vtvm3616_TRIG_SRC_INT_CH1       Internal Channel 1
                  vtvm3616_TRIG_SRC_INT_CH2       Internal Channel 2
                  vtvm3616_TRIG_SRC_INT_CH3       Internal Channel 3
                  vtvm3616_TRIG_SRC_INT_CH4       Internal Channel 4
                  vtvm3616_TRIG_SRC_INT_CH5       Internal Channel 5
                  vtvm3616_TRIG_SRC_INT_CH6       Internal Channel 6
                  vtvm3616_TRIG_SRC_INT_CH7       Internal Channel 7
                  vtvm3616_TRIG_SRC_INT_CH8       Internal Channel 8
                  vtvm3616_TRIG_SRC_INT_CH9       Internal Channel 9
                  vtvm3616_TRIG_SRC_INT_CH10      Internal Channel 10
                  vtvm3616_TRIG_SRC_INT_CH11      Internal Channel 11
                  vtvm3616_TRIG_SRC_INT_CH12      Internal Channel 12
                  vtvm3616_TRIG_SRC_INT_CH13      Internal Channel 13
                  vtvm3616_TRIG_SRC_INT_CH14      Internal Channel 14
                  vtvm3616_TRIG_SRC_INT_CH15      Internal Channel 15
                  vtvm3616_TRIG_SRC_INT_CH16      Internal Channel 16

ViBoolean   slope,
                  - This parameter is used to set the slope of the Trigger
                  selected. The Slope of the trigger may be Positive /
                  Negative. Selects which edge of the triggering signal is the
                  active edge. The slope applies only to the external and  TTL
                  trigger sources i.e., the valid values of  trigger source
                  for which the slope can be  configured are mentioned below:
```

```
                  vtvm3616_TRIG_SRC_EXT
                  vtvm3616_TRIG_SRC_TTLTRG0
                  vtvm3616_TRIG_SRC_TTLTRG1
                  vtvm3616_TRIG_SRC_TTLTRG2
                  vtvm3616_TRIG_SRC_TTLTRG3
                  vtvm3616_TRIG_SRC_TTLTRG4
                  vtvm3616_TRIG_SRC_TTLTRG5
                  vtvm3616_TRIG_SRC_TTLTRG6
                  vtvm3616_TRIG_SRC_TTLTRG7

Valid Range:      vtvm3616_SLOPE_POSITIVE (or)
                  vtvm3616_SLOPE_NEGATIVE.

ViReal32 voltLevelValue,
                  - This parameter is used to set the Voltage Level.

Valid Range:
                  vtvm3616_VOLT_LEVEL_VALUE_MIN (-20.00000) to
                  vtvm3616_VOLT_LEVEL_VALUE_MAX (+19.99939).

ViInt16     channelList[],
                  - This parameter specifies the channel list for which the
                  specified voltage level is to be set.

Valid Range for each channel:
                  vtvm3616_CHANNEL_LIST_MIN           (1) to
                  vtvm3616_CHANNEL_LIST_VM3608_MAX (8) for 3608 module
                  vtvm3616_CHANNEL_LIST_MIN (1) to
                  vtvm3616_CHANNEL_LIST_VM3616_MAX (16) for 3616 module

                  ViInt16     numOfChannels,
                  - This parameter is used to set the number of channels to be
                  configured.

Valid Range:      vtvm3616_CHANNEL_LIST_MIN  (1) to
                  vtvm3616_CHANNEL_LIST_VM3608_MAX  (8) for 3608 module

                  vtvm3616_CHANNEL_LIST_MIN  (1) to
                  vtvm3616_CHANNEL_LIST_VM3616_MAX 16) for 3616 module

Return Values:    Returns VI_SUCCESS if successful.
                  Else returns error value.

Description       This is an application function which shows how to group
                  core driver functions to setup trigger source and output
                  voltage level for a selected list of channels.
                  Please look at the source code of this function to help you
                  understand how to group core driver functions in your
                  application.
*************************************************************************/
```

```
ViStatus _VI_FUNC vtvm3616_setupAndWriteToDAC(ViSession instrHndl,
                  ViInt16     triggerSource,
                  ViBoolean   slope,
                  ViReal32    voltLevelValue,
                  ViInt16     channelList[],
                  ViInt16     numOfChannels)
{
      ViStatus    status      = VI_NULL;

status = vtvm3616_configTriggerParams (instrHndl, triggerSource, slope);
      if (status < VI_SUCCESS)
            return vtvm3616_ERROR_SETTING_TRIGGER_PARAMS;

            status = vtvm3616_setupVoltage(instrHndl,
                  voltLevelValue,
                  channelList,
                  numOfChannels);
      if (status < VI_SUCCESS)
            return vtvm3616_ERROR_SETTING_OUTPUT_VOLTAGE;

            return VI_SUCCESS;
}

/**************************************************************************
Function    :           vtvm3616_setupScanListParams

Formal Parameters ViSession   instrHndl,
                        - A valid session handle to the instrument.

ViInt16     channel,
                        - This parameter specifies the channel for which  the
                        scan-list operation mode is to be set.

Valid Range:            vtvm3616_CHANNEL_LIST_MIN (1) to
                        vtvm3616_CHANNEL_LIST_VM3608_MAX (8) if it is VM3608.
                                          or
                        vtvm3616_CHANNEL_LIST_MIN (1) to
                        vtvm3616_CHANNEL_LIST_VM3616_MAX (16)  if it is
                        VM3616.

ViInt16     mode,
                        - This parameter specifies the mode of operation to be
                        set for the scan list operation of the specified
                        channel.

Valid Range: Interpretation:
                        vtvm3616_SCAN_MODE_OFF  Scan Mode Off.
                        vtvm3616_SCAN_MODE_ON   Scan Mode On.
                        vtvm3616_SCAN_MODE_LOOP Scan Mode Loop.
```

```
ViInt16     count,
                        - This parameter specifies the position in the scan
                        list array where the interrupt routine
                        loading the DACs should either stop or loop back to
                        zero.

Valid Range:            vtvm3616_COUNT_MIN (1)  to
                        vtvm3616_COUNT_MAX (512).


ViReal32 voltageList[],
                        - This parameter specifies the voltage values  to be
                        set in the scan list arrays of the specified channel.

                        Each element of the array should be of the range
                        specified below:

                        vtvm3616_VOLT_LEVEL_VALUE_MIN (-20.00000) to
                        vtvm3616_VOLT_LEVEL_VALUE_MAX (19.99939).

ViInt16     numOfVolts
                        - This parameter specifies the number of valid entries
                        for the voltage list array.

Valid Range:            vtvm3616_VOLTAGE_LIST_MIN (1)  to
                        vtvm3616_VOLTAGE_LIST_MAX (512).

Return Values:           Returns VI_SUCCESS if successful.
                        Else returns error value.

Description             This function is an application function that shows
                        how the user can use core functions to load the
                        specified channel's scan list with the voltage values.
                        These values can then be loaded to the DAC using the
                        interrupt routine.
*************************************************************************/
ViStatus _VI_FUNC vtvm3616_setupScanListParams( ViSession instrHndl,
ViInt16 channel,
ViInt16 mode,
ViInt16 count,
ViReal32voltageList[],
ViInt16     numOfVolts)
{
ViStatus status = VI_NULL;
ViInt16 channelList[1];

      channelList[0] = channel;
/*
 * Resetting the module to its default state
 */
      status = vtvm3616_reset(instrHndl);
      if (status < VI_SUCCESS)
            return status;

/*
 * Configuring the TTLT Line 0 as the trigger source for the
 * Scan Mode
 */
      status = vtvm3616_configTriggerParams (instrHndl,
                  vtvm3616_TRIG_SRC_TTLTRG0,
                  vtvm3616_SLOPE_POSITIVE);
      if (status < VI_SUCCESS)
            return vtvm3616_ERROR_SETTING_TRIGGER_PARAMS;

      /*
       * Configuring the Scan Mode of the specified channel
       */
      status = vtvm3616_setupScanMode(instrHndl,mode,channelList,1);

      if (status < VI_SUCCESS)
            return vtvm3616_ERROR_SETTING_SCAN_MODE;

      /*
       * Configuring the Scan Limit Index in the scan array where the
       * interrupt routine loading the DAC will either stop or loop
       * back to zero depending on the scan mode
       */
status = vtvm3616_setupScanLimit(instrHndl, channel, count);
      if (status < VI_SUCCESS)
            return vtvm3616_ERROR_SETTING_SCAN_LIMIT;


      /*
       * Configuring the Scan List for the specified channel
       */
status = vtvm3616_setupScanList(instrHndl,      channel, voltageList,
numOfVolts);
      if (status < VI_SUCCESS)
            return vtvm3616_ERROR_SETTING_SCAN_LIST;

      return VI_SUCCESS;
}
```

SECTION 4

COMMAND DICTIONARY

INTRODUCTION

This section presents the instrument command set. It begins with an alphabetical list of all the commands supported by the VM3608A/3616A are divided into three sections: IEEE 488.2 commands, the instrument specific SCPI commands and the required SCPI commands. With each command is a brief description of its function, whether the command's value is affected by the *RST command and its default value.

The remainder of this section is devoted to describing each command, one per page, in detail. The description is presented in a regular and orthogonal way assisting the user in the use of each command. Every command entry describes the exact command and query syntax, the use and range of parameters and a complete description of the command's purpose.

ALPHABETICAL COMMAND LISTING

The following tables provide an alphabetical listing of each command supported by the VM3608A/3616A along with a brief description. If an X is found in the column titled *RST, then the value or setting controlled by this command is possibly changed by the execution of the *RST command. If no X is found, then *RST has no effect. The default column gives the value of each command's setting when the unit is powered up or when a *RST command is executed.

TABLE 4-1: IEEE 488.2 COMMON COMMANDS

Command	Description	*RST	Reset Value
*CLS	Clears the Status Register.	X	
*ESE	Sets the Event Status Enable Register.	X	
*ESR?	Query the Standard Event Status Register		N/A
*IDN?	Query the module identification string.		N/A
*OPC	Set the OPC bit in the Event Status Register		N/A
*RST	Resets the module to a known state		N/A
*SRE	Set the service request enable register		N/A
*STB?	Query the Status Byte Register.		N/A
*TRG	Causes a trigger event to occur.		N/A
*TST?	Starts and reports a self-test procedure.		N/A
*WAI	Halts execution and queries	X	

TABLE 4-2: INSTRUMENT SPECIFIC SCPI COMMANDS

Command	Description	*RST	Reset Value
ABORt	Stop current operation		N/A
CALibration:COUNt?	Returns a number that indicates the number of times the VM3608A/3616A has been calibrated.		N/A
CALibration:DATA	Manually sets or queries the calibration constants.	X	Values from non-volatile
CALibration:GAIN	Used to set the calibration constant for the gain of the selected channel; its effect are immediate.	X	Values from non-volatile
CALibration:SECure:CODE	Sets the code required to disable calibration security.		N/A
CALibration:SECure:STATe	Enables or disables calibration security.	X	1 (security enabled)
CALibration:STORe	Saves the current calibration constants into non-volatile memory.		N/A
CALibration:STORe:AUTO	Allows the new calibration constants to be saved to non-volatile memory as they are changed by the CALibration:GAIN and CALibration:ZERO commands	X	1
CALibration:ZERO	Used to set the calibration constant for the offset of the selected channel.	X	Values from non-volatile
FORMat	Sets the output format for digital queries.	X	ASCII/ (Decimal)
INITiate	Arms unit for operation in ARB and FIFO mode		N/A
MEMory:SETup	Enters a voltage list for manual loading to the Precision DACs.	X	0
MEMory:SIZE?	Queries the RAM memory size		N/A
OUTPut:TTLTrg	Sets output VXIbus trigger line	X	0
OUTPut:TTLTrg:SOURce	Sets outputs trigger signal source	X	NONE
SCAN	Enables/disables the operation of the scan list function.	X	Off
SCAN OFF	Sets SCAN mode to OFF on all channels		N/A
SCAN:LIMit	Sets the scan loop-back limit.	X	512
SCAN:TABLe	Enters a voltage list to be placed in the scan list of the specified channel.	X	0
SCAN:TABLe:LOCation	Enters a voltage in a specific location in the list.		N/A
SOURce:MARKer:SYNC:SLOPe	Sets the marker at a positive or negative level	X	POS
SOURce:MARKer:SYNC:STATe	Enables or disables the front panel marker output	X	0
SOURce:MARKer:TRIGger	Enables or disables the marker trigger		N/A
SOURce:ROSCillator:SLOPe	Selects Reference OSCillator slope	X	POS
SOURce:ROSCillator:SOURce	Selects the source of the ARB and FIFO sample check	X	TIM
SOURce:SEQuence:ADVance	Selects how a sequence advances	X	Sync

Instrument Specific SCPI Commands (continued)

Command	Description	*RST	Reset Value
SOURce:SEQuence:DWEL1	Sets the number of times to loop through a sequence segment		N/A
SOURce:SEQuence:LENGth	Sets the number of segments in a sequence list		N/A
SOURce:SEQuence:LIST	Sets up the segment sequence list		N/A
SOURce:SEQuence:STARt	Selects between Automatic or Triggered starting of a segment		N/A
SOURce:VOLTage:DATA	Sets the output level of the channels selected by the channel list using the binary data programmed into the 16-bit DAC.	X	0 V
SOURce:VOLTage:FORMat	Set the data format for voltage levels	X	1
SOURce:VOLTage:LEVel	Sets the output level of the channels selected by the channel list.	X	0 V
SOURce:VOLTage:RANGe	Sets the output voltage range.	X	20 V
SOURce:VOLTage:SETup	Sets the output level of all channels using the binary data from a selected location in the scan list.		N/A
TRACe:DATA	Loads data into the selected Trace memory		N/A
TRACe:DATA:POINt	Sets the value of the data point in Trace memory		N/A
TRACe:DEFine	Sets the segment number and its size		N/A
TRACe:DELete:ALL	Deletes all Trace definitions		N/A
TRACe:FREE?	Returns the number of Trace points used and the number of points available		N/A
TRACe:LEVel	Sets the voltage level		N/A
TRACe:POINts	Re-sizes a Trace segment		N/A
TRIGger:IMMediate	Command immediately triggers the instrument		N/A
TRIGger:IMMiadiate:ADVance	Sets the trigger for advancing segments		N/A
TRIGger:SLOPe	Selects which edge of a triggering signal is the active edge.	X	Positive edge
TRIGger:SOURce	Selects the trigger event which updates the output DACs	X	AUTO mode
TRIGger:TIMer	Sets the time interval for the internal periodic trigger source	X	1e-3

TABLE 4-3: SCPI REQUIRED COMMANDS

Command	Description	*RST	Reset Value
STATus:OPERation:CONDition?	Queries the Operation Status Condition Register.	X	
STATus:OPERation:ENABle	Sets the Operation Status Enable Register.	X	
STATus:OPERation[:EVENt]?	Queries the Operation Status Event Register.	X	
STATus:PRESet	Presets the Status Register.	X	
STATus:QUEStionable:CONDition?	Queries the Questionable Status Condition Register	X	
STATus:QUEStionable:ENABle	Sets the Questionable Status Enable Register.	X	
STATus:QUEStionable[:EVENt]?	Queries the Questionable Status Event Register	X	
SYSTem:ERRor?	Queries the Error Queue	X	Clears queue
SYSTem:VERsion?	Queries the version of the SCPI standard for which the module complies.		N/A

COMMAND DICTIONARY

The remainder of this section is devoted to the actual command dictionary. Each command is fully described on its own page. In defining how each command is used, the following items are described:

Purpose	Describes the purpose of the command.
Type	Describes the type of event, such as type or setting.
Command Syntax	Details the exact command format
Command Parameters	Describes the parameters sent with the command and their legal parameters
***RST Value**	Describes the value assumed when the *RST (reset) command is sent.
Query Syntax	Details the exact query form of the command.
Query Parameters	Describes the parameters sent with the command and their legal range. The default parameter values are assumed the same as in the command form unless described otherwise.
Query Response	Describes the format of the query response and the valid range of output.
Description	Describes in detail what the command does and refers to additional sources.
Examples	Presents the proper use of each command and its query (when available).
Related Commands	Lists commands that affect the use of this command or commands that are affected by this command.

IEEE 488.2 Common Commands

*CLS

Purpose	Clears the Status Register.	
Type	IEEE 488.2 Common Command	
Command Syntax	*CLS	
Command Parameters	None	
***RST Value**	N/A	
Query Syntax	None	
Query Parameters	N/A	
Query Response	N/A	
Description	This command clears all event registers, clears the OPC flag and clears all queues (except the output queue).	
Examples	**Command / Query**	**Response *(Description)***
	`*CLS`	
Related Commands	None	

*ESE

Purpose	Sets the bits of the Event Status Enable Register.	
Type	IEEE 488.2 Common Command	
Command Syntax	*ESE <mask>	
Command Parameters	<mask> = numeric ASCII value	
***RST Value**	None, the parameter is required	
Query Syntax	*ESE?	
Query Parameters	None	
Query Response	Numeric ASCII value from 0 to 255	
Description	The Event Status Enable command is used to set the bits of the Event Status Enable Register. See ANSI/IEEE488.2-1987 section 11.5.1 for a complete description of the ESE register. A value of 1 in a bit position of the ESE register enables generation of the ESB (Event Status Bit) in the Status Byte by the corresponding bit in the ESR. If the ESB is set in the SRE register then an interrupt will be generated. See the ESR? command for details regarding the individual bits. The ESE register layout is: Bit 0 - Operation Complete Bit 1 - Request Control (not used in the VM3608A/3616A) Bit 2 - Query Error Bit 3 - Device Dependent Error (not used in the VM3608A/3616A) Bit 4 - Execution Error Bit 5 - Command Error Bit 6 - User Request (not used in the VM3608A/3616A) Bit 7 - Power On The Event Status Enable query reports the current contents of the Event Status Enable Register.	
Examples	**Command / Query**	**Response *(Description)***

	`*ESE 36`
	`*ESE?`
		
		36

Related Commands	*ESR?	

*ESR?

Purpose	Queries and clears the Standard Event Status Register.
Type	IEEE 488.2 Common Command
Command Syntax	None - Query only
Command Parameters	N/A
***RST Value**	N/A
Query Syntax	ESR?
Query Parameters	None
Query Response	Numeric ASCII value from 0 to 255
Description	The Event Status Register query - queries and clears the contents of the Standard Event Status Register. This register is used in conjunction with the ESE register to generate the ESB (Event Status Bit) in the Status Byte. The layout of the ESR is: Bit 0 - Operation Complete Bit 1 - Request Control (not used in the VM3608A/3616A, always 0) Bit 2 - Query Error Bit 3 - Device Dependent Error (not used in the VM3608A/3616A, always 0) Bit 4 - Execution Error Bit 5 - Command Error Bit 6 - User Request (not used in the VM3608A/3616A, always 0) Bit 7 - Power On The Operation Complete bit is set by the VM3608A/3616A when it receives an *OPC command. The Query Error bit is set when data is over-written in the output queue. This could occur if one query is followed by another without reading the data from the first query. The Execution Error bit is set when an execution error is detected. Errors that range from -200 to -299 are execution errors. The Command Error bit is set when a command error is detected. Errors that range from -100 to -199 are command errors. The Power On bit is set when the module is first powered on or after it receives a reset via the VXI Control Register. Once the bit is cleared (by executing the *ESR? command) it will remain cleared.
Examples	**Command / Query**: `*ESR?` — **Response *(Description)***: 4
Related Commands	*ESE <mask>

*IDN?

Purpose	Queries the module for its identification string.	
Type	IEEE 488.2 Common Command	
Command Syntax	None - Query Only	
Command Parameters	N/A	
***RST Value**	N/A	
Query Syntax	*IDN?	
Query Parameters	None	
Query Response	ASCII character string	
Description	The Identification query returns the identification string of the VM3608A/3616A module. The response is divided into four fields separated by commas. The first field is the manufacturer’s name, the second field is the model number, the third field is an optional serial number and the fourth field is the firmware revision number. If a serial number is not supplied, the third field is set to 0 (zero).	
Examples	**Command / Query**	**Response *(Description)***
	`*IDN?`	VXI Technology Inc.,VM3616A,0,1.1
Related Commands	None	

*OPC

Purpose	Sets the OPC bit in the Event Status Register.
Type	IEEE 488.2 Common Command
Command Syntax	*OPC
Command Parameters	None
***RST Value**	N/A
Query Syntax	*OPC?
Query Parameters	None
Query Response	1
Description	The Operation Complete command sets the OPC bit in the Event Status Register when all pending operations have completed. The Operation Complete query will return a 1 to the output queue when all pending operations have completed.

Examples

Command / Query	**Response *(Description)***
`*OPC`	
`*OPC?`	1

Related Commands	*WAI

*RST

Purpose	Resets the module's hardware and software to a known state.	
Type	IEEE 488.2 Common Command	
Command Syntax	*RST	
Command Parameters	None	
***RST Value**	N/A	
Query Syntax	None	
Query Parameters	N/A	
Query Response	N/A	
Description	The Reset command resets the module's hardware and software to a known state. See the command index at the beginning of this chapter for the default parameter values used with this command.	
Examples	**Command / Query**	**Response *(Description)***
	`*RST`	
Related Commands	None	

*SRE

Purpose	Sets the service request enable register.
Type	IEEE 488.2 Common Command
Command Syntax	*SRE <mask>
Command Parameters	<mask> = Numeric ASCII value from 0 to 255
***RST Value**	None – required parameter
Query Syntax	*SRE?
Query Parameters	None
Query Response	Numeric ASCII value from 0 to 255
Description	The service request enable mask is used to control which bits in the status byte generate back plane interrupts. If a bit is set in the mask that newly enables a bit set in the status byte and interrupts are enabled, the module will generate a REQUEST TRUE event via an interrupt. See the *STB? Command for the layout of bits. **Note:** Bit 6 is always internally cleared to zero as required by IEEE 488.2 section 11.3.2.3. The layout of the Service Request Enable Register is: Bit 0 - Unused Bit 1 - Unused Bit 2 - Error Queue Has Data Bit 3 - Questionable Status Summary (not used) Bit 4 - Message Available Bit 5 - Event Status Summary Bit 6 - 0 Bit 7 - Operation Status Summary

Examples	**Command / Query**	**Response *(Description)***
	`*SRE 4`	
	`*SRE?`	4
Related Commands	None	

*STB?

Purpose	Queries the Status Byte Register.	
Type	IEEE 488.2 Common Command	
Command Syntax	None - Query Only	
Command Parameters	N/A	
***RST Value**	N/A	
Query Syntax	*STB?	
Query Parameters	None	
Query Response	Numeric ASCII value from 0 to 255	
Description	The Read Status Byte query fetches the current contents of the Status Byte Register. See the IEEE 488.2 specification for additional information regarding the Status byte Register and its use. The layout of the Status Register is: Bit 0 - Unused Bit 1 - Unused Bit 2 - Error Queue Has Data Bit 4 - Questionable Status Summary (not used) Bit 5 - Message Available Bit 6 - Master Summary Status Bit 7 - Operation Status Summary	
Examples	**Command / Query**	**Response *(Description)***
	`*STB?`	16
Related Commands	None	

*TRG

Purpose	Causes a trigger event to occur.	
Type	IEEE 488.2 Common Command	
Command Syntax	*TRG	
Command Parameters	None	
***RST Value**	N/A	
Query Syntax	None	
Query Parameters	N/A	
Query Response	N/A	
Description	The Trigger command causes a trigger event to occur. In the VM3608A/3616A this is used to start transmitting the data in all the queues whose associated channel is in BLOCK MODE. See the section discussing block mode of operation for further details.	
Examples	**Command / Query**	**Response *(Description)***
	`*TRG`	
Related Commands	None	

*TST?

Purpose

Causes a self-test procedure to occur and queries the results.

Type

IEEE 488.2 Common Command

Command Syntax

None - Query Only

Command Parameters

N/A

***RST Value**

N/A

Query Syntax

*TST?

Query Parameters

None

Query Response

Numeric ASCII value from 0 to 143

Description

The Self-Test query causes the VM3608A/3616A to run its self-test procedures and report on the results. The following tests are performed:

1. Each channel runs an internal loop-back self-test.
2. The buffer RAM runs a simple self-test.

The *TST? query returns a numeric ASCII from 0 to 143 which has the following reading:

Bit 0	Channel 1 Failed	
Bit 1	Channel 2 Failed	
Bit 2	Channel 3 Failed	
Bit 3	Channel 4 Failed	
Bit 4	Unused	Databus bit failure to FPGA
Bit 5	Unused	
Bit 6	Unused	
Bit 7	RAM Test Failed	
Bit 8 – 14	Code to indicate a RAM Failure (Address or Data)	
Bit 15	RAM Flood Test Failure	

A bit value of 1 in any location indicates a failure while a 0 value indicates that the test passed. The RAM test failed bit indicates that the buffer RAM used for the data queues failed to pass a simple pseudo random patter test or an all zeros test.

Examples

Command / Query	**Response *(Description)***
`*TST?`	0

Related Commands

None

*WAI

Purpose	Halts execution of additional commands and queries until the No Operation Pending message is true.	
Type	IEEE 488.2 Common Command	
Command Syntax	*WAI	
Command Parameters	None	
***RST Value**	N/A	
Query Syntax	None	
Query Parameters	N/A	
Query Response	N/A	
Description	The Wait to Continue command halts the execution of commands and queries until the No Operation Pending message is true. This command makes sure that all previous commands have been executed before proceeding. It provides a way of synchronizing the module with its commander.	
Examples	**Command / Query**	**Response *(Description)***
	`*WAI`	
Related Commands	*OPC	

DEVICE SPECIFIC SCPI COMMANDS

ABORt

Purpose	Stop current block operations.	
Type	Device dependent SCPI command	
Command Syntax	ABORt	
Command Parameters	None	
***RST Value**	None	
Query Syntax	None – Command Only	
Query Parameters	N/A	
Query Response	N/A	
Description	This command stops the current block operations and all active timers; the buffers and settings are unchanged. This command is an event and has no associated, no query form and no *RST value.	
Examples	**Command / Query**	**Response *(Description)***
	`ABOR`	
Related Commands	None	

CALibration:COUNt?

Purpose	Returns a number that indicates the number of times the VM3608A/3616A has been calibrated.	
Type	Query	
Command Syntax	None - Query Only	
Command Parameters	N/A	
***RST Value**	N/A	
Query Syntax	CALibration:COUNt?	
Query Parameters	None	
Query Response	Numeric ASCII value. The maximum value for the count is 16,777,215 after which it will wrap to 0	
Description	The Calibration Count query returns the number of times the VM3608A/3616A has been calibrated. The instrument will increment the count every time the non-volatile memory storing the calibration constants is updated. If the calibration security is disabled (CALibration:SECure:STATe OFF is set) and CALibration:STORe:AUTO ON is set, the count will increment with each execution of the CALibration:GAIN or CALibration:ZERO command. If the CALibration:STORe:AUTO OFF is set, the count will only be incremented by invoking the CALibration:STORe command. The non-volatile memory has a guaranteed minimum of 10,000 cycles.	
Examples	**Command / Query**	**Response *(Description)***
	`CAL:COUN?`	3
Related Commands	CALibration:STORe	

CALibration:DATA

Purpose	Manually sets or queries the calibration constants.
Type	Setting
Command Syntax	CALibration:DATA <block_data>
Command Parameters	<block_data> = Data in IEEE 488.2 definite or indefinite length arbitrary block format
***RST Value**	Set to values stored in non-volatile memory
Query Syntax	CALibration:DATA?
Query Parameters	None
Query Response	Returns the calibration constants in IEEE 488.2 definite or indefinite length arbitrary block format
Description	The calibration constants are implemented as 8-bit values from -127 to +128. The constants are treated as an ordered set with the following indices: The Calibration Data command is used to set calibration constants in the VM3608A/ VM3616A. The constants will change only if the calibration security is disabled. New constants take effect immediately, but are not saved to non-volatile memory unless the CALibration:STORe command is executed regardless if CALibration:STORe:AUTO is ON or OFF. The whole data set must be provided or an error will be generated. The calibration constants set or queried apply to the currently set range for each channel, therefore a mixture of 10 V and 20 V range constants may be loaded or retrieved.
Examples	**Command / Query** `CAL:DATA #232 12300174011021230014367192100156`
Related Commands	CALibration<channel>:GAIN<value>, CALibration<channel>:ZERO<value> CALibration:STORe

Index	Contents	Index	Contents
0	Channel 1 gain	1	Channel 2 gain
2	Channel 3 gain	3	Channel 4 gain
4	Channel 5 gain	5	Channel 6 gain
6	Channel 7 gain	7	Channel 8 gain
8	Channel 9 gain	9	Channel 10 gain
10	Channel 11 gain	11	Channel 12 gain
12	Channel 13 gain	13	Channel 14 gain
14	Channel 15 gain	15	Channel 16 gain
16	Channel 1 offset	17	Channel 2 offset
18	Channel 3 offset	19	Channel 4 offset
20	Channel 5 offset	21	Channel 6 offset
22	Channel 7 offset	23	Channel 8 offset
24	Channel 9 offset	25	Channel 10 offset
26	Channel 11 offset	27	Channel 12 offset
28	Channel 13 offset	29	Channel 14 offset
30	Channel 15 offset	31	Channel 16 offset

CALibration:GAIN

Purpose	Manually sets the calibration constant for the gain of the selected channel.
Type	Setting
Command Syntax	CALibration<channel>:GAIN<value>
Command Parameters	<channel> =1 - 16 or 1 - 8 referring to a specific Calibration DAC <value> =The range for value is -128 to +127
***RST Value**	Set to values stored in non-volatile memory.
Query Syntax	CALibration<channel>:GAIN?
Query Parameters	<channel> = 1 - 16 referring to a specific Calibration DAC
Query Response	<value> = -128 to +127 in the specified format
Description	The Calibration Gain command sets the calibration constant for the gain of the selected channel. Its effect is immediate. If the CALibration:STORe:AUTO ON is set, the command will save the new constant to the non-volatile memory each time the command is set. If the CALibration:STORe:AUTO OFF is set, a CALibration:STORe command must be executed in order to save the new constant. The Calibration Gain command will function only when the calibration security is disabled, otherwise an error is generated. The query returns the value from the non-volatile memory rather than the currently used value and may be different from the constant currently being used.
Examples	
Related Commands	CALibration<channel>:ZERO<value>, CALibration:DATA <block_data>, CALibration:STORe, CALibration:STORe:AUTO <mode>, FORMat<type>

Command / Query	Response *(Description)*
`CAL2:GAIN -120`	
`CAL2:GAIN?`	-120

CALibration:SECure:CODE

Purpose	Sets the code required to disable the calibration security.	
Type	Setting	
Command Syntax	CALibration:SECure:CODE <code>	
Command Parameters	<code> = an ASCII sring 1 to 12 characters in length entered in IEEE 488.2 definite or indefinite length arbitrary block format	
***RST Value**	N/A	
Query Syntax	CALibration:SECure:CODE?	
Query Parameters	N/A	
Query Response	IEEE 488.2 definite length arbitrary block	
Description	The Calibration Secure Code command sets the code required to disable the calibration security. Calibration security must be disabled in order to change the code string. Before shipping, instruments are factory set the code to ‘VM3616’ for a VM3616A and ‘VM3608’ for a VM3608A. The Query Only works if calibration security has been previously disabled. Note that the security code is case sensitive.	
Examples	**Command / Query**	**Response *(Description)***
	CAL:SEC:CODE #17VM3616A	
	CAL:SEC:CODE?	#17VM3616A
Related Commands	CALibration:SECureSTATe <mode>[,<code>]	

CALibration:SECure:STATe

Purpose	Enables or disables calibration security.	
Type	Setting	
Command Syntax	CALibration:SECure:STATe <boolean>,<code>	
Command Parameters	<boolean> = 0 | 1 | OFF | ON <code> = Security code in IEEE 488.2 definite length arbitrary block format.	
***RST Value**	1 or ON	
Query Syntax	CALibration:SECure:STATe?	
Query Parameters	None	
Query Response	<mode> = 1 | 0	
Description	The Calibration Security State command enables or disables the calibration security. When the security state is ON or 1, the calibration constants may not changed or stored to the non-volatile memory. To change and/or store new calibration constants to the non-volatile memory, the Security State must be OFF or 0. In order to disable the Security State, the security code must be supplied and must be in IEEE-488.2 definite length arbitrary block format. The code parameter must be present to disable the security or it generates error –109, "Missing parameter". The value must match the currently programmed security code or it generates error –224, "Illegal parameter value". To enable security, the code parameter is not required, but if it is provided, it will be checked. If the code is given but does not match the current security code, error –224, "Illegal parameter value" will be generated. The query returns the current mode.	
Examples	**Command / Query**	**Response *(Description)***
	CAL:SEC:STAT OFF,#15OLIVE	*(The above says to turn the security state off so that the calibration constant can be stored in non-volatile. The password is assumed to be OLIVE. Note the password is case sensitive.)*
	CAL:SEC:STAT?	0 *(The above says that the security state is presently off.)*
	CAL:SEC:STAT ON	*(The above says to turn the security state on so that calibration constants cannot be stored in non-volatile.)*
	CAL:SEC:STAT?	ON *(The above says that the security state is presently on.)*
Related Commands	CALibration:SECure:CODE<code>, CALibration:STORe	

CALibration:STORe

Purpose	Saves the current calibration constants into the non-volatile memory.	
Type	Setting	
Command Syntax	CALibration:STORe	
Command Parameters	None	
***RST Value**	N/A	
Query Syntax	N/A	
Query Parameters	N/A	
Query Response	N/A	
Description	The Calibration Store command saves the current calibration constants into the non-volatile memory. The CALibration:SECure:STATe must be OFF before using this command.	
Examples	**Command / Query**	**Response *(Description)***
	CAL:STOR	*(Saves the current calibration constants to non-volatile memory.)*
Related Commands	CALibration<channel>:GAIN<value>, CALibration<channel>:ZERO<value>, CALibration:DATA <block_data>, CALibration:COUNt ?	

CALibration:STORe:AUTO

Purpose	Allows the new calibration constants to be saved to non-volatile memory, automatically.
Type	Setting
Command Syntax	CALibration:STORe:AUTO<mode>
Command Parameters	<mode> = 0 | OFF | 1 | ON. The command will not execute if calibration security is enabled and error -203, "Command protected" is generated.
***RST Value**	ON
Query Syntax	CALibration:STORe:AUTO?
Query Parameters	None
Query Response	<mode> = 1 | 0. 1 means the auto store is enabled. 0 means the auto store is disabled.
Description	The Calibration Store Auto command allows the new calibration constants to be saved to non-volatile memory, automatically. The new calibration constants are automatically saved to non-volatile memory as they are changed by the CALibration:GAIN and CALibration:ZERO commands, but not the CALibration:DATA command.
Examples	*This example assumes SECURITY is disabled.*
Related Commands	CALibration<channel>:GAIN<value> CALibration<channel>:ZERO<value>

Examples:

Command / Query	**Response *(Description)***
CAL:STOR:AUTO ON	
CAL:STOR:AUTO?	1

CALibration:ZERO

Purpose	Used to set the calibration constant for the offset of the selected channel; its effect is immediate.
Type	Setting
Command Syntax	CALibration<channel>:ZERO<value>
Command Parameters	<channel> = 1 - 16 or 1 - 8 referring to a specific calibration DAC <value> = –128 to +127.
***RST Value**	Set to values stored in the non-volatile memory.
Query Syntax	CALibration<channel>:ZERO?
Query Parameters	<channel> Selects a specific DAC, 1 of 8 for the VM3608A, 1 of 16 for the VM3616A. Specify 1 to select the first channel.
Query Response	<value> = -128 to +127 in the specified format
Description	The Calibration Zero command is used to set the calibration constant for the offset of the selected channel; its effect is immediate. If the CALibration:STORe:AUTO ON is set, the command will save the new constant to the non-volatile memory each time the command is set. If the CALibration:STORe:AUTO OFF is set, a CALibration:STORe command must be executed in order to save the new constant. The CALibration:ZERO command will function only when calibration security is disabled, otherwise an error is generated. The query returns the currently used calibration value for the selected channel(s), in the format specified by the FORMat <type> command.
Examples	
Related Commands	CALibration<channel>:GAIN<value>, CALibration:DATA <block_data>, CALibration:STORe, CALibration:STORe:AUTO<mode>, FORMat <type>

Command / Query	**Response *(Description)***
CAL1:ZERO 115	
CAL1:ZERO?	115
FORM HEX	
CAL1:ZERO?	#H73
FORM OCT	
CAL1:ZERO?	#Q163
FORM:BIN	
CAL1:ZERO?	#B1110011

FORMat

Purpose	Sets the output format for the digital queries.	
Type	Setting	
Command Syntax	FORMat <type>	
Command Parameters	<type> = ASCii, HEXadecimal, OCTal, BINary	
***RST Value**	ASCii	
Query Syntax	FORMat?	
Query Parameters	None	
Query Response	<type> = ASC | HEX | OCT | BIN	
Description	The Format command sets the format of the returned data from the instrument. ASCII specifies numbers expressed in decimal. Leading zeros are suppressed. HEXadecimal expresses numbers in a 2-digit leading 0 alphanumeric format. Numbers A - F are in capitals. OCTal expresses numbers in a 3-digit leading 0 format. BINary expressed numbers in an 8-digit leading 0 format. The query returns the output format for the digital queries.	
Examples	**Command / Query**	**Response *(Description)***
	FORM ASC	*(Sets the output data to be in ASCII format)*
	FORM?	ASC (*Queries then reports the output format is ASCII)*
Related Commands	CALibration<channel>:GAIN<value>, CALibration<channel>:ZERO<value>	

INITiate

Purpose	Sets unit into waiting-for-trigger-state.	
Type	Event	
Command Syntax	INITiate:[IMMediate]	
Command Parameters	None	
***RST Value**	N/A	
Query Syntax	N/A	
Query Parameters	N/A	
Query Response	N/A	
Description	The Initiate Immediate command arms the VM3608A/3616A upon receipt of the command.	
Examples	**Command / Query**	**Response *(Description)***
	INIT:IMM	
Related Commands	ABORt	

MEMory:SETup

Purpose	To allow the entry of a separate voltage list this will be loaded to the Precision DACs upon command.	
Type	Setting	
Command Syntax	MEMory:SETup <index>,<voltage_list>	
Command Parameters	<index> = integer number from 1 to 512 (which specifies the array element) <voltage_list> = a list of 8 (for the VM3608A) or 16 (for the VM3616A) voltages	
***RST Value**	0 (all elements in the memory array are set to 0 volts)	
Query Syntax	MEMory:SETup? <index>	
Query Parameters	<index> = 1 - 512 (which specifies the array element)	
Query Response	The query form of this command returns the voltages at position <index> for all DACs in their respective 512-element array. The format is a set of voltages delimited by commas.	
Description	The Memory Setup command allows the entry of a separate voltage list that will be loaded to the Precision DACs. Each DAC channel has an associated 512-element "memory" array. The same elements in all 16 arrays are loaded at the same time from the supplied 16-element voltage list.	
Examples	**Command / Query**	**Response *(Description)***
	MEM:SET 1, 2,3,4,5,2,3,4,5,2,3,4,5,2,3, 4,5	
	MEM:SET? 1	2.000122,2.999878,4.000244,5.000000, 2.000122,2.999878,4.000244,5.000000, 2.000122,2.999878,4.000244,5.000000, 2.000122,2.999878,4.000244,5.000000
Related Commands	SOURce:VOLTage:SETup <index>	

MEMory:SIZE?

Purpose	Queries the RAM memory size.	
Type	Query	
Command Syntax	None	
Command Parameters	N/A	
***RST Value**	N/A	
Query Syntax	MEMory:SIZE?	
Query Parameters	None	
Query Response	524288 or 1048576	
Description	The Memory Size query inquires the amount of RAM on each unit.	
Examples	**Command / Query**	**Response *(Description)***
	MEM:SIZE?	524288
Related Commands	N/A	

OUTPut:TTLTrg

Purpose	Selects which line to use	
Type	Setting	
Command Syntax	OUTPut:TTLTrg <trigline>	
Command Parameters	<trigline> = 0 to 7	
***RST Value**	0	
Query Syntax	OUTPut:TTLTrg?	
Query Parameters	None	
Query Response	0 to 7	
Description	The OUTPut:TTLTrg command sets which VXIbus TTL trigger line is used.	
Examples	**Command / Query**	**Response *(Description)***
	OUTP:TTLT 3	
	OUTP:TTLT?	3
Related Commands	OUTPut:TTLTrg:SOURce	

OUTPut:TTLTrg:SOURce

Purpose	Sets the output trigger source.	
Type	Setting	
Command Syntax	OUTPut:TTLTrg:SOURce <trigsrc>	
Command Parameters	<trigsrc> = NONE | TRIGGER | MARKER	
***RST Value**	NONE	
Query Syntax	OUTPut:TTLTrg:SOURce?	
Query Parameters	N/A	
Query Response	NONE | TRIGGER | MARKER	
Description	This command set the output trigger source to the VXIbus trigger lines: **NONE** There is no signal to the VXIbus TTL. **TRIGGER** The internal trigger signal is output to the VXIbus TTL. **MARKER** The set MARKERS are output to the VXIbus TTL. (Note that the MARKER setting option is only available in **ARB** mode.)	
Examples	**Command / Query**	**Response *(Description)***
	OUTP:TTLT:SOUR MARKER	
Related Commands	OUTPut:TTLTrg <trigline>	

SCAN

Purpose	Enables/disables the operation of the scan list operation for the specified channels
Type	Setting
Command Syntax	SCAN <mode>,<channel_list>
Command Parameters	<mode> = 0 | OFF |1 | ON | LOOP | FIFO | ARBitrary <channel_list> = A list of DAC channels in channel list format
***RST Value**	0
Query Syntax	SCAN? <channel>
Query Parameters	<channel> = channel number 1 - 16 or 1 - 8, specifying a specific DAC
Query Response	<mode> = 0 | 1 | LOOP | FIFO | ARB
Description	The Scan command enables or disables the operation of the scan list operation for the specified channels. For each channel that is enabled, the interrupt routine will load a voltage from its respective scan list arrays at the current array position to the DAC and auto increment the scan list array pointer. If the array pointer equals the limit, then the scan function for that channel will stop, unless the mode of that channel is set to LOOP. LOOP mode means the scan function will reset the array pointer to 0 and continue. When the scan mode is to be enabled, the trigger source of either EXTernal or one of the TTLTriggers is required to be selected first. In ARBitrary mode, the TRIGger:IMMediate:ADVance must also be set. **Note:** Legal transitions of SCAN modes are: OFF ⇨ ON OFF ⇨ FIFO ON ⇨ OFF FIFO ⇨ OFF OFF ⇨ LOOP OFF ⇨ ARB LOOP ⇨ OFF ARB ⇨ OFF Example illegal transitions of SCAN modes are: LOOP ⇨ ON ON ⇨ FIFO ARB ⇨ LOOP The channel's Scan mode setting must pass through the OFF setting first before being set to any other mode. Scan modes OFF, ON, and LOOP may be simultaneously assigned to different channels. The FIFO and ARB modes may not be simultaneously assigned with any other modes other than OFF. **NOTE:** In FIFO and ARB mode, DACs in the OFF mode cannot be changed. If they need to be changed, all DACs must first be removed from FIFO and ARB mode. For example: Channels 1 and 4 may be set to ARB while the rest are OFF; or Channels 1 through 6 may be set to FIFO while the rest are OFF. The SCAN OFF command is useful as it sets all channel modes OFF simultaneously so that they may be reassigned.
Examples	**Command / Query** | **Response *(Description)*** SCAN ON,(@1:4) | SCAN? 2 | 1
Related Commands	SCAN:LIMit <channel>,<count>, SCAN:TABLe <channel>,<voltage list>, SCAN OFF

SCAN OFF

Purpose	Sets Scan mode to OFF on all channels	
Type	Setting	
Command Syntax	SCAN OFF	
Command Parameters	None	
***RST Value**	OFF	
Query Syntax	None	
Query Parameters	N/A	
Query Response	N/A	
Description	Sets SCAN mode to OFF on all channels. This feature is useful for resetting the scan mode on all channels before reassigning them. See **SCAN**.	
Examples	**Command / Query**	**Response *(Description)***
	SCAN OFF	
Related Commands	SCAN <mode>,<channel_list>	

SCAN:LIMit

Purpose	Sets the point in the 512-element scan list array where the scanning should either stop or loop back to zero.	
Type	Setting	
Command Syntax	SCAN:LIMit <channel>,<count>	
Command Parameters	<channel> = A specific DAC channel of 1 - 16 or 1 - 8 <count> = A position [ranging from 1 - 512] in the array to stop or loop back	
***RST Value**	512	
Query Syntax	SCAN:LIMit ? <channel>	
Query Parameters	<channel> = A specific DAC channel of 1 - 16 or 1 - 8	
Query Response	<count> = 1 - 512	
Description	The Scan Limit command specifies a position in the 512 element scan list array where the interrupt routine loading the DACs should either stop or loop back to 0, depending on the configured scan mode.	
Examples	**Command / Query**	**Response *(Description)***
	SCAN:LIM 2,256	
	SCAN:LIM? 2	256
Related Commands	SCAN, SCAN:TABle:LOCation <number>,<voltage>, SCAN:TABLe	

SCAN:TABLe

Purpose	Enters a list of voltages to be placed in the specified channel's scan list.
Type	Setting
Command Syntax	SCAN:TABLe <channel>,<voltage_list>
Command Parameters	<channel> = A specific DAC channel of 1 - 16 or 1 - 8 <voltage_list> = A list of voltages to be loaded in this channel's scan list. These are voltage values delimited by commas. The number of voltages in the list ranges from 1 to 512.
***RST Value**	0 (all elements in the array are set to 0 volts)
Query Syntax	SCAN:TABLe ? <channel> [count [,start]]
Query Parameters	<channel> = A specific DAC channel of 1 - 16 or 1 - 8 <count> = The number of voltages to be returned. If not specified the entire 512 element scan list will be returned <start> = Specifies a point in the 512 element array to begin the returning of voltages.
Query Response	A list of voltages delimited by commas according to the <count> and <start> parameters. If count and start are not specified the entire 512 element scan list will be returned.
Description	The Scan Table command loads a scan list of a specific channel with voltage values. These values are then loaded to the DAC from the interrupt routine. This operation is dependent upon the scan mode and the scan limit. Each channel has its own independent mode, limit, scan list array and pointer in the array (some channels could loop back while others continue in the array). The instrument does not have an on-board clock. So, in order to set up the scan mode, the trigger source that triggers through the scan list must also be specified. The valid trigger sources for the SCAN mode are TTLT and EXT.
Examples	**Command / Query** — **Response *(Description)***
	SCAN:TABL 1,2,3,4,5 —
	SCAN:TABL? 1 3,2 — 2.999878,4.000244,5.000000
Related Commands	SCAN <mode>,<channel_list>, SCAN:LIMit <channel> <count>, TRIGger:SOURce <source>

SCAN:TABLe:LOCation

Purpose	Enters a voltage at a specific location in a scan list array of a channel.	
Type	Setting	
Command Syntax	SCAN:TABLe<channel>:LOCation <number>,<voltage>	
Command Parameters	<channel> = A specific DAC channel of 1 - 16 or 1 - 8 <number> = A specific location in the scan list array ranging between 1 - 512 <voltage> = A single voltage value ranging between -20.00000 and +19.99939	
***RST Value**	N/A	
Query Syntax	SCAN:TABLe<channel>:LOCation? <number>	
Query Parameters	<channel> = A specific DAC channel of 1 - 16 or 1 - 8. <number> = A specific location in the scan list array between 1 and 512.	
Query Response	Returns the voltage value from the scan list of a specified channel at the specified location.	
Description	The Scan Table Location command allows a voltage value at a specific location in the scan list to be modified or queried.	
Examples	**Command / Query**	**Response *(Description)***
	SCAN:TABL1:LOC 2,4	
	SCAN:TABL1:LOC? 2	4.000244
Related Commands	SCAN <mode>,<channel_list>, SCAN:TABLe <channel>,<voltage_list>	

SOURce:MARKer:SYNC:SLOPe

Purpose	Sets the marker at a positive or negative level.	
Type	Setting	
Command Syntax	SOURce:MARKer:SYNC:SLOPe <slope>	
Command Parameters	<slope> = POSitive | NEGative	
***RST Value**	POS	
Query Syntax	SOURce:MARKer:SYNC:SLOPe?	
Query Parameters	None	
Query Response	POS | NEG	
Description	Set the marker at a positive or negative level pulse. Note that the MARKER function is only available in ARBitrary mode.	
Examples	**Command / Query**	**Response *(Description)***
	SOUR:MARK:SYNC:SLOP POS	
Related Commands	SOURce:MARKer:TRIGger	

SOURce:MARKer:SYNC:STATe

Purpose	Enables or disables the front panel marker output.	
Type	Setting	
Command Syntax	SOURce:MARKer:SYNC:STATe <boolean>	
Command Parameters	<boolean> = 0 | OFF | 1 | ON	
***RST Value**	0	
Query Syntax	SOURce:MARKer:SYNC:STATe?	
Query Parameters	N/A	
Query Response	0 | 1	
Description	The Source Marker Sync State command enables or disables the front-panel marker output. The power up and reset condition is with the output disabled. If a parameter of 1 or ON is sent with this command, the marker output will be enabled and will generate marker pulses when the instrument is in ARBitrary waveform generation mode. If a parameter of 0 or OFF is sent with this command, the marker pulse will not be generated and the marker output will remain at a logic low level. Note that the output is still driven.	
Examples	**Command / Query**	**Response *(Description)***
	SOUR:MARK:SYNC:STAT ON	
	SOUR:MARK:SYNC:STAT?	1
Related Commands	SOURce:MARKer:SYNC:STATe SOURce:MARKer:TRIGger	

SOURce:MARKer:TRIGger

Purpose	Enables or disables the marker trigger	
Type	Setting	
Command Syntax	SOURce:MARKer:TRIGger[:STATe] <boolean>,<seq_index>	
Command Parameters	<boolean> = 0 | OFF | 1 | ON <seq_index> = segment index number (or pointer) from 0 to 4095 in the channel list format	
***RST Value**	N/A	
Query Syntax	SOURce:MARKer:TRIGger[:STATe]?	
Query Parameters	<seq_index> = segment index number (or pointer) from 0 to 4095	
Query Response	0 | 1	
Description	The Source Marker Trigger function enables or disables the trigger pulse that marks the beginning of each segment within a sequence. This function is only available in ARBitrary mode.	
Examples	**Command / Query**	**Response *(Description)***
	SOUR:MARK:TRIG 1,(@20)	
	SOUR:MARK:TRIG? 20	1
Related Commands	SOURce:MARKer:SYNC:SLOPe	

SOURce:ROSCillator:SLOPe

Purpose	Selects the active slope for advancing data when in FIFO or ARBitrary waveform generation mode.	
Type	Setting	
Command Syntax	SOURce:ROSCillator:SLOPe <slope>	
Command Parameters	<slope> = POSitive | NEGative.	
***RST Value**	<slope> = POS	
Query Syntax	SOURce:ROSCillator:SLOPe?	
Query Parameters	N/A	
Query Response	POS | NEG	
Description	The Source Roscillator Slope command selects the active edge for advancing data when in FIFO, or ARBitrary waveform generation mode. This Command Only applies to the front panel clock input and to the TTL trigger bus inputs. If the positive edge is selected, a rising edge on the clock signal will cause the data to advance. If the negative edge is selected, a falling edge will advance the data. If the timer or internal clock source is selected, this command will have no effect.	
Examples	**Command / Query**	**Response *(Description)***
	SOUR:ROSC:SLOP NEG	*(Selects the falling edge)*
	SOUR:ROSC:SLOP?	NEG *(Queries then reports that the current setting is a negative slope.)*
Related Commands		

SOURce:ROSCillator:SOURce

Purpose	Selects the source oscillator for pacing data in FIFO or ARBitrary waveform generation mode.	
Type	Setting	
Command Syntax	SOURce:ROSCillator:SOURce <source>	
Command Parameters	<source> = INTernal | EXTernal | EXTCLK | TIMer | TTLTrg<n> (*where n = 0 – 7)*	
***RST Value**	<source> = TIM	
Query Syntax	SOURce:ROSCillator:SOURce?	
Query Parameters	N/A	
Query Response	INT | EXT | EXTCLK | TIM | TTLT<n> (*where n = 0 – 7)*	
Description	The Source Roscillator Source command selects which event will cause new data to update the active DACs when the instrument is in either FIFO mode or ARBitrary waveform generation mode. The selected source establishes the data output sample rate for the instrument. The sources are defined as follows: INTernal: Selects software command for pacing data out. The data is advanced upon receipt of a TRIGger:IMMediate or *TRG command. EXTernal CLOCK: Selects the front panel trigger input for pacing data output. The data is advanced upon receipt of the rising or falling edge, as selected. TIMer: Selects the internal timer for pacing data out. The data is advanced at the interval programmed into the timer. TTLTrig<n>: Selects one of the eight VXIbus TTL trigger lines for pacing data output. The data is advanced upon receipt of a rising or falling edge, as selected.	
Examples	**Command / Query**	**Response *(Description)***
	SOUR:ROSC:SOUR TTLT5	
	SOUR:ROSC:SOUR?	TTLT5
Related Commands	SOURce:ROSCillator:SLOPe	

SOURce:SEQuence:ADVance

Purpose	Selects how a sequence advances from segment to segment.	
Type	Setting	
Command Syntax	[SOURce:]SEQuence:ADVance <type>,<seq_index>	
Command Parameters	<type> = SYNChronous | ASYNchronous <seq_index> = Numeric value from 0 to 4095	
***RST Value**	SYNChronous	
Query Syntax	[SOURce:]SEQuence:ADVance? <seq_index>	
Query Parameters	<seq_index> = numeric value from 0 to 4095	
Query Response	SYNC | ASYN	
Description	The Source Sequence Advance command selects how a sequence advances from one segment the next when a trigger is required to initiate an advance. The two types of advance function as follows: SYNChronous Advances to the next segment after a trigger is received and the current segment has completed. ASYNchronous Advances to the next segment upon receipt of a trigger without waiting for the current segment to complete. The sequence index parameter selects which one or more of 4096 possible sequences, or segments, to apply the command to. Only one channel may be queried at a time.	
Examples	**Command / Query**	**Response *(Description)***
	SEQ:ADV ASYN,(@1:5)	*(Sets the second through the 6th segment for asynchronous advance.)*
	SOUR:SEQ:ADV? 5	ASYN *(Queries the advance type for the 6th segment)*
Related Commands	N/A	

SOURce:SEQuence:DWELl

Purpose	Sets the number of times to loop through a segment in a sequence.
Type	Setting
Command Syntax	[SOURce:]SEQuence:DWEL1 <numeric_value>,<seq_index>
Command Parameters	<numeric_value> = numeric value from 0 to 1048575 <seq_index> = numeric value from 0 to 4095
***RST Value**	All segments set to 1
Query Syntax	[SOURce:]SEQuence:DWEL1? <seq_index>
Query Parameters	<seq_index> = a numeric value from 0 to 4095
Query Response	A numeric value from 0 to 1048575
Description	The Source Sequence Dwell command sets the number of time a sequence will loop through a specific segment before moving on the next segment. The loop count can range from 1 to 1048575. If a value of 0 is programmed, the segment will loop indefinitely. A triggered advance should be setup to advance out of a segment with a loop count of 0. The sequence index parameter selects which one or more of 4096 possible segments to apply the command to. Only one channel may be queried at a time.
Examples	**Command / Query** — **Response *(Description)*** SEQ:DWEL 10,(@10:25) SEQ:DWEL? 20 — 10
Related Commands	N/A

SOURce:SEQuence:LENGth

Purpose	Sets the number of segments in a sequence list.	
Type	Setting	
Command Syntax	[SOURce:]SEQuence:LENGth <numeric_value>	
Command Parameters	<numeric_value> = A numeric value from 1 to 4096	
***RST Value**	1	
Query Syntax	[SOURce:]SEQuence:LENGth?	
Query Parameters	N/A	
Query Response	Numeric value from 1 to 4096	
Description	The Source Sequence Length command sets the number of segments in a sequence list. A maximum of 4096 segments may be defined or a minimum of 1 may be used.	
Examples	**Command / Query**	**Response *(Description)***
	SEQ:LENG 20	
	SEQ:LENG?	20
Related Commands	SOURce:SEQuence:LIST	

SOURce:SEQuence:LIST

Purpose	Sets up the segment sequence list.	
Type	Setting	
Command Syntax	[SOURce:]SEQuence:LIST <seq_list>	
Command Parameters	<seq_list> = a series of numeric values from 0 to 4095 in standard channel list format	
***RST Value**	1	
Query Syntax	[SOURce:]SEQuence:LIST?	
Query Parameters	N/A	
Query Response	a series of numeric values from 0 to 4095 in standard channel list format	
Description	The Source Sequence List command sets the order of segments to create the sequence list. A maximum of 4096 segments may be defined or a minimum of 1 may be used.	
Examples	**Command / Query**	**Response *(Description)***
	SEQ:LIST (@2,2,3,4,2,2)	
	SEQ:LIST?	2,2,3,4,2,2
Related Commands	SOURce:SEQuence:LIST	

SOURce:SEQuence:STARt

Purpose	Selects between automatic or triggered starting of a segment in a sequence.
Type	Setting
Command Syntax	[SOURce:]SEQuence:STARt <type>,<seq_index>
Command Parameters	<type> = AUTOmatic | TRIGgered <seq_index> = a series of numeric values from 0 to 4095 in standard channel list format
***RST Value**	Automatic
Query Syntax	[SOURce:]SEQuence:STARt? <seq_index>
Query Parameters	<seq_index> = Numeric value from 0 to 4095
Query Response	AUTO | TRIG
Description	The Source Sequence Start command selects between automatic or triggered segment advancement. When a segment has completed its run it will advance to the next segment in the following ways: Automatic Start causes a new segment to begin as soon as the previous segment completes. There is no external trigger required to cause this to happen. Triggered Start causes a completed segment to hold its last output value until a trigger is received to advance to the new segment. The sequence index parameter selects which one or more of 4096 possible segments to apply the command to. Only one channel may be queried at a time
Examples	**Command / Query** — SEQ:STAR TRIG,(@1:15) SEQ:STAR TRIG? 12 **Response *(Description)*** — TRIG
Related Commands	N/A

Examples:

Command / Query	Response *(Description)*
SEQ:STAR TRIG,(@1:15)	
SEQ:STAR TRIG? 12	TRIG

SOURce:VOLTage:DATA

Purpose	Sets the output level of the channels selected using the specified data
Type	Setting
Command Syntax	[SOURce:]VOLTage:DATA <value>,<channel_list>
Command Parameters	<value> = the range entered in set format (see description) <channel_list> = a list of channels to be loaded with the specified value
***RST Value**	Sets all channels to 0 volts
Query Syntax	[SOURce:]VOLTage:DATA? <channel>
Query Parameters	<channel> = this parameter specifies which Precision DAC, the query is targeting
Query Response	<value> = -32768 to +32767, i.e., -20.00000 to +19.99939 or -10 to 9.9999695 in Decimal
Description	The Source Voltage Data command sets the output level of the channels selected by the channel list using the specified data. The 16-bit precision DAC is programmed with the Decimal, HEX, Octal, or Binary value specified.

Example using the 20 V Range:

With the SOURce:VOLTage:FORMat set to OFF:

Voltage	Decimal	Hex	Octal	Binary
-20 V	0	#H0	#Q0	#B0
0 V	32768	#H8000	#Q100000	#B1000000000000000
+20 V	65535	#HFFFF	#Q177777	#B1111111111111111

With SOURce:VOLTage:FORMat set to ON:

Voltage	Decimal	Hex	Octal	Binary
-20 V	-32768	#H8000	#Q100000	#B1000000000000000
-0.00061 V	-1	#HFFFF	#Q177777	#B1111111111111111
0 V	0	#H0	#Q100000	#B0
+20 V	32767	#H7FFF	#Q077777	#B0111111111111111

The query command reports the Decimal value of one channel at a time in the same format as the command.

Examples

Command / Query	**Response *(Description)***
SOUR:VOLT:DATA 16384,(@1,2,3)	*(Loads Channels 1, 2 ,3.)*
SOUR:VOLT:DATA? 3	16384

Related Commands SOURce:VOLTage:FORMat, SOURce:VOLTage:LEVel

SOURce:VOLTage:FORMat

Purpose	Set the data format for voltage levels	
Type	Setting	
Command Syntax	SOURce:VOLTage:FORMat <boolean>	
Command Parameters	<boolean> = 0 | OFF | 1 | ON	
***RST Value**	1	
Query Syntax	SOURce:VOLTage:FORMat?	
Query Parameters	None	
Query Response	0 | 1	
Description	The Voltage Format command sets the format for the voltage data as implemented in the word serial command SOURce:VOLTage:DATA, and in Register Access. Note that Register Access and Word Serial are the opposite of each other. See the *Register Access* section for Register Access details. For Word Serial, when this setting is 0, or OFF, the binary voltage inputs are **#H0 to #H8000 = -<range> to 0 V**, and **#H8000 to #HFFFF = 0 to (<range> - LSB)**. For Word Serial, when this setting is 1, or ON, the binary voltage inputs are **#H0 to #H7FFF = 0 to (<range> - LSB)**, and **#H8000 to #HFFFF = -<range> to (0 - LSB)**; *where <range> is **10v for the 10 V range and 20v for the 20 V range**, and **LSB = (2 * <range>) / 65536**.* See ***Programming the DACs via Register Access*** in Section 3.	
Examples	**Command / Query**	**Response *(Description)***
	SOUR:VOLT:FORM 1	*(Sets the binary voltage data format to #H0 to #H7FFF = positive voltage levels and #H8000 to #HFFFF = negative voltage levels.)*
Related Commands	SOURce:VOLTage:DATA	

SOURce:VOLTage:LEVel

Purpose	Sets the output voltage level of the channels selected by the channel list	
Type	Setting	
Command Syntax	[SOURce:]VOLTage[:LEVel] <value>,<channel_list>	
Command Parameters	<value> = the range for the value parameter is from -20.00000 to +19.99939 <channel_list> = channel list from 1 through 16 or 1 through 8 to be loaded with a specified value	
***RST Value**	Sets all channels to 0 volts	
Query Syntax	[SOURce:]VOLTage[:LEVel]? <channel>	
Query Parameters	<channel> = this parameter specifies which Precision DAC, the query is targeting	
Query Response	<value> = 20.00000 to +19.99939	
Description	The Source Voltage Level command sets the output voltage level of channels selected by the channel. The Voltage input parameter is converted to a 16-bit binary representation used to program the 16-bit precision DAC. The query command reports the value of one channel at a time in the same format as the command.	
Examples	**Command / Query**	**Response *(Description)***
	SOUR:VOLT:LEV -10,(@1,2,3)	
	SOUR:VOLT:LEV? 3	-10.000000
Related Commands	SOURce:VOLTage:DATA <value>,<channel_list>	

SOURce:VOLTage:RANGe

Purpose	Loads each DAC with a voltage value from the location in its respective "memory lists" specified by <index>.	
Type	Event	
Command Syntax	SOURce:VOLTage:RANGe <range>,<channel_list>	
Command Parameters	<range> = 10v | 20v <channel_list> = Standard format channel list spanning Channels 1 through 16 for the VM3616A, or Channels 1 through 8 for the VM3608A	
***RST Value**	20v	
Query Syntax	SOURce:VOLTage:RANGe? <channel>	
Query Parameters	<channel> = 1 to 16 for the VM3616A or 1 to 8 for the VM3608A.	
Query Response	10v | 20v	
Description	The Source Voltage Range command selects the output voltage range of the VM3608A/3616A DAC. In the 10V range, the output spans from +9.99969 V to -10.00000 V and has a resolution of 305 µV. In the 20 V range, the output spans from +19.99939 V to -20.00000 V and has a resolution of 610 µV.	
Examples	**Command / Query**	**Response *(Description)***
	SOUR:VOLT:RANG 10V, (@1,2,3)	*(Sets the voltage range for Channels 1 through 3 to 10 V.)*
Related Commands	N/A	

SOURce:VOLTage:SETup

Purpose	Loads each DAC with a voltage value from the location in its respective "memory lists" specified by <index>.	
Type	Setting	
Command Syntax	SOURce:VOLTage:SETup <index>	
Command Parameters	<index> = a number from 1 to 512	
***RST Value**	N/A	
Query Syntax	None	
Query Parameters	None	
Query Response	None	
Description	The Source Voltage Setup command loads the DAC with voltages at a specified index from the memory list rather than from an embedded value in an instrument SCPI command such as SOURce:VOLTage:DATA or from the scan list which loads from the interrupt routine. The memory list is set up using the MEMory:SETup command.	
Examples	**Command / Query**	**Response *(Description)***
	MEM:SET 1, 2,3,4,5	
	SOUR:VOLT:LEV? 1	2.000122
	SOUR:VOLT:LEV? 2	2.999878
	SOUR:VOLT:LEV? 3	4.000244
	SOUR:VOLT:LEV? 4	5.000000
Related Commands	MEMory:SETup <index>,<voltage_list>	

TRACe:DATA

Purpose	Loads data into the selected trace memory.	
Type	Setting	
Command Syntax	TRACe:DATA <trace_index>,<channel>,<block> ***or*** TRACe:DATA <trace_index>,<channel>,<trace_index>,<channel>	
Command Parameters	<trace_index>= Integer value from 0 to 4095 <channel> = 1 through 8 or 1 through 16 <block> = Data in definite length arbitrary block data format	
***RST Value**	All points have a value of 0	
Query Syntax	TRACe:DATA? <trace_index>	
Query Parameters	<trace_index> = Integer value from 0 to 4095	
Query Response	Definite length arbitrary block data format	
Description	The Trace Data command loads data into a specific trace memory block. Each trace memory block is identified by a trace index. There may be as many as 4096 traces, or as few as one. If a trace index and a definite length arbitrary block of data is passed with the command, the data is stored to trace memory. If two trace index values are passed with the command, the data stored in the second trace index location is copied to the first trace index location. If the source trace is longer than the destination trace, the data at the end of the source trace is not used. If the source trace data is shorter that the destination, all excess locations in the destination trace are loaded with zeros. **Note**: Valid only in ARBitrary mode.	
Examples	**Command / Query**	**Response *(Description)***
	TRAC:DATA 17,2,#13234567	*(Assigns a block of data to trace 17 on Channel 3.)*
	TRAC:DATA? 17,3	#16234567
	TRAC:DATA 4090,4,17,3	(*Copies the data from trace 17 Channel 3 to trace 4090 Channel 4.)*
	TRAC:DATA? 4090,4	#16234567
Related Commands	N/A	

TRACe:DATA:POINt

Purpose	Sets a value at a selected point in trace memory.	
Type	Setting	
Command Syntax	TRACe:DATA:POINt <trace_index>,<arb_channel>,<trace_point>,<volt_data>	
Command Parameters	<trace_index> = 0 to 4095 <arb_channel> = The channel on which the data is to be modified. <trace_point> = Either 1 to 524288 or 1 to 1048576 (minus 28 k) <volt_data> = Data value for the desired voltage level.	
***RST Value**	All points are set to 0	
Query Syntax	TRACe:DATA:POINt? <trace_index>,<arb_channel>,<trace_point>	
Query Parameters	<trace_index> = 0 to 4095 <arb_channel> = The channel on which the data is to be modified <trace_point> = Either 1 to 524288 or 1 to 1048576 (minus 28 k)	
Query Response	0 - 65535 representing the voltage data at the queried point.	
Description	The Trace Data Point command sets a specific output value to a specific location in trace memory. **Note**: Valid only in ARBitrary mode.	
Examples	**Command / Query**	**Response *(Description)***
	TRAC:DATA:POIN 0,1,100,2000	
	TRAC:DATA:POIN? 0,1,100	2000
Related Commands	TRACe:DATA TRACe:LEVel	

TRACe:DEFine

Purpose	Sets the segment number and its size.	
Type	Setting	
Command Syntax	TRACe:DEFine <trace_index>,<trace_size>	
Command Parameters	<trace_index> = 0 to 4095 <trace_size> = Either 1 to 524288 or 1 to 1048576 (minus 28 k)	
***RST Value**	N/A	
Query Syntax	None	
Query Parameters	N/A	
Query Response	N/A	
Description	The Trace Define command set the trace index number and the trace size. **Note**: Valid only in ARBitrary mode.	
Examples	**Command / Query**	**Response *(Description)***
	TRAC:DEF 1,20	
Related Commands	N/A	

TRACe:DELete:ALL

Purpose	Deletes the data from the traces.	
Type	Operation	
Command Syntax	TRACe:DELete:ALL	
Command Parameters	None	
***RST Value**	N/A	
Query Syntax	None	
Query Parameters	N/A	
Query Response	N/A	
Description	The Trace Delete ALL command deletes all data from the traces and resizes them to contain 2 elements. The two remaining elements are set to a value of 0. The trace index is still valid, but all memory previously allocated to the trace except for two words is freed. **Note**: Valid only in ARBitrary mode	
Examples	**Command / Query**	**Response *(Description)***
	TRAC:DEL:ALL	
Related Commands	N/A	

TRACe:FREE?

Purpose	Query that returns number of trace points used and available.	
Type	Query	
Command Syntax	N/A	
Command Parameters	N/A	
***RST Value**	N/A	
Query Syntax	TRACe:FREE?	
Query Parameters	N/A	
Query Response	Numeric value available, numeric value used	
Description	The Trace Free command queries and reports the available trace points. The first number represents the amount available, and the second is the amount used. **Note**: Valid only in ARBitrary mode.	
Examples	**Command / Query**	**Response *(Description)***
	TRAC:FREE?	494616,1000
Related Commands	N/A	

TRACe:LEVel

Purpose	Sets the voltage level.	
Type	Setting	
Command Syntax	TRACe:LEVel <trace_index>,<channel>,<volt_list>	
Command Parameters	<trace_index> = 0 to 4095 <channel> = 1 through 8 or 1 through 16 <volt_list> = separated voltage level list within the set range	
***RST Value**	N/A	
Query Syntax	TRACe:LEVel? <trace_index>,<channel>	
Query Parameters	<trace_index> = 0 to 4095 <channel> = 1 through 8 or 1 through 16	
Query Response	<volt_list> = comma separated voltage level list within the set range	
Description	The Trace Level command sets the trace voltage levels at the specified channel. **Note**: Valid only in ARBitrary mode.	
Examples	**Command / Query**	**Response *(Description)***
	TRAC:LEV 0,1,-10,-9,-8	*(Sets the voltage level list for trace 0, Channel 1.)*
	TRAC:LEV? 0,1	-10.000000,-9.000000,-8.000000
Related Commands	N/A	

TRACe:POINts

Purpose	Resizes a trace segment length.	
Type	Setting	
Command Syntax	TRACe:POINts <trace_index>,<trace_points>	
Command Parameters	<trace_index> = 0 to 4095 <trace_points> = Either 1 to 524288 or 1 to 1048576 (minus 28 k)	
***RST Value**	All trace segments are sized to two points each	
Query Syntax	TRACe:POINts? <trace_index>	
Query Parameters	<trace_index> = 0 to 4095	
Query Response	Either 1 to 524288 or 1 to 1048576 (minus 28 k)	
Description	The Trace Points command sets the length (number of data points) of a specified trace. The number of trace points may only be resized to a number less than or equal to the initial setting. **Note**: Valid only in ARBitrary mode.	
Examples	**Command / Query**	**Response *(Description)***
	TRAC:POIN 0,48	*(Resets the points at trace 0 to 48.)*
Related Commands	N/A	

TRIGger:IMMediate

Purpose	Causes a trigger event to occur.	
Type	Event	
Command Syntax	TRIGger:IMMediate	
Command Parameters	N/A	
***RST Value**	N/A	
Query Syntax	N/A	
Query Parameters	N/A	
Query Response	N/A	
Description	Causes a trigger event to occur.	
Examples	**Command / Query**	**Response *(Description)***
	TRIG:IMM	
Related Commands	TRIGger:SOURce	

TRIGger:IMMediate:ADVance

Purpose	Sets the trigger for the reference oscillator.	
Type	Setting	
Command Syntax	TRIGger:IMMediate:ADVance	
Command Parameters	None	
***RST Value**	N/A	
Query Syntax	N/A	
Query Parameters	N/A	
Query Response	N/A	
Description	Creates an event for the advance signal to advance data (data update). Valid only for FIFO and ARBitrary modes.	
Examples	**Command / Query**	**Response *(Description)***
	TRIG:IMM:ADV	
Related Commands	N/A	

TRIGger:SLOPe

Purpose	Selects the active trigger edge.	
Type	Setting	
Command Syntax	TRIGger:SLOPe <slope>	
Command Parameters	<slope> = POSitive | NEGative	
***RST Value**	Positive	
Query Syntax	TRIGger:SLOPe?	
Query Parameters	N/A	
Query Response	POS or NEG	
Description	The Trigger Slope Command Only applies to the External and TTL trigger sources. It selects which edge of a triggering signal is the active edge. The query reports the edge that was selected.	
Examples	**Command / Query**	**Response *(Description)***
	TRIG:SLOP POS	
	TRIG:SLOP?	POS
Related Commands	TRIGger:SOURce <source>	

TRIGger:SOURce

Purpose	Selects the trigger event which updates the DACs.
Type	Setting
Command Syntax	TRIGger:SOURce <source>
Command Parameters	NONE – Used to turn off other trigger sources so that a TRIGger IMMediate command may be used. INTernal <channel> = All outputs are updated when the selected channel is updated (all other channels wait for the selected channel to update). AUTO = A DAC output is immediately updated when the channel is programmed. The second event is not required to update the output voltage EXTernal = Selects the front panel trigger input and all channels are updated (all other channels wait for the selected channel to update). TTLTrig<n> = All outputs update when the selected TTL trigger line goes active.
***RST Value**	AUTO
Query Syntax	TRIGger:SOURce?
Query Parameters	None
Query Response	<source> = AUTO | EXT | INT 1 - 16 | TTLT 0 - 7 (for VM3616A) <source> = AUTO | EXT | INT 1 - 8 | TTLT 0 - 7 (for VM3608A)
Description	The Trigger Source command is used to select the source that updates the DACs on the VM3608A/3616A. Each DAC is double buffered. Therefore, writing to the DAC (source:voltage:data or source:voltage:level) will require a second event to cause the output voltage to be updated. This command selects the source of the update event. When using SCAN, a trigger source of either EXTernal or one of the TTLTriggers is required. This command is used to select that source. The query reports the source of the update event that was selected.
Examples	**Command / Query**: TRIG:SOUR TTLT3 — **Response *(Description)***: **Command / Query**: TRIG:SOUR? — **Response *(Description)***: TTLT3
Related Commands	TRIGger:SLOPe <slope>, SCAN:TABLe <channel>,<voltage_list>

TRIGger:TIMer

Purpose	Sets the period of the internal timer.	
Type	Setting	
Command Syntax	TRIGger:TIMer <period>	
Command Parameters	<period> = 1.0e-5 to 4.2949e2	
***RST Value**	1.0e-3	
Query Syntax	TRIGger:TIMer?	
Query Parameters	N/A	
Query Response	Numeric value ranging from 1.0e-5 to 4.2949e2	
Description	Sets the period of the internal timer. The value can be set in 100 ns increments. The internal timer is one source for the advance update.	
Examples	**Command / Query**	**Response *(Description)***
	TIM 1.0E-5	
Related Commands	N/A	

REQUIRED SCPI COMMANDS

STATus:OPERation:CONDition?

Purpose	Queries the Operation Status Condition Register.	
Type	Required SCPI command	
Command Syntax	None – Query Only	
Command Parameters	N/A	
***RST Value**	N/A	
Query Syntax	STATus:OPERation:CONDition?	
Query Parameters	None	
Query Response	0	
Description	The Operation Status Condition Register query is provided for SCPI compliance only. The VM3608A/3616A does not alter the state of any of the bits in this register and always reports a 0.	
Examples	**Command / Query**	**Response *(Description)***
	STAT:OPER:COND?	
Related Commands	None.	

STATus:OPERation:ENABle

Purpose	Sets the Operation Status Enable Register.	
Type	Required SCPI command	
Command Syntax	STATus:OPERation:ENABle <NRf>	
Command Parameters	<NRf> = numeric ASCII value from 0 to 32767	
***RST Value**	<NRf> must be specified	
Query Syntax	STATus:OPERation:ENABle?	
Query Parameters	None	
Query Response	<NRf> = Numeric ASCII value from 0 to 32767	
Description	The Operation Status Enable Register is included for SCPI compatibility and the VM3608A/3616A does not alter any of the bits in this register. The register layout is as follows: Bit 0 – Calibrating Bit 1 – Setting Bit 2 – Ranging Bit 3 – Sweeping Bit 4 – Measuring Bit 5 – Waiting for trigger Bit 6 – Waiting for arm Bit 7 – Correcting	
Examples	**Command / Query**	**Response *(Description)***
	STAT:OPER:ENAB 0 STAT:OPER:ENAB?	 0
Related Commands	None	

STATus:OPERation:EVENt?

Purpose	Queries the Operation Status Event Register.	
Type	Required SCPI command	
Command Syntax	None - Query Only	
Command Parameters	N/A	
***RST Value**	N/A	
Query Syntax	STATus:OPERation [:EVENt]?	
Query Parameters	None	
Query Response	0	
Description	The Status Operation Event Register query is included for SCPI compliance. The VM3608A/3616A does not alter any of the bits in this register and always reports a 0.	
Examples	**Command / Query**	**Response *(Description)***
	STAT:OPER?	0
Related Commands	None	

STATus:PRESet

Purpose	Presets the Status Registers	
Type	Required SCPI command	
Command Syntax	STATus:PRESet	
Command Parameters	None	
***RST Value**	N/A	
Query Syntax	None – Command Only	
Query Parameters	N/A	
Query Response	N/A	
Description	The Status Preset command presets the Status Registers. The Operational Status Enable Register is set to 0 and the Questionable Status Enable Register is set to 0. This command is provided for SCPI compliance only.	
Examples	**Command / Query**	**Response *(Description)***
	STAT:PRES	
Related Commands	None	

STATus:QUEStionable:CONDition?

Purpose	Queries the Questionable Status Condition Register.	
Type	Required SCPI command	
Command Syntax	None – Query Only	
Command Parameters	N/A	
***RST Value**	N/A	
Query Syntax	STATus:QUEStionable:CONDition?	
Query Parameters	None	
Query Response	0	
Description	The Questionable Status Condition Register query is provided for SCPI compliance only. The VM3608A/3616A does not alter any of the bits in this register and a query always reports a 0.	
Examples	**Command / Query**	**Response *(Description)***
	STAT:QUES:COND?	0
Related Commands	None	

STATus:QUEStionable:ENABle

Purpose	Sets the Questionable Status Enable Register.	
Type	Required SCPI command	
Command Syntax	STATus:QUEStionable:ENABle <NRf>	
Command Parameters	<NRf> = numeric ASCII value from 0 to 32767	
***RST Value**	<NRf> must be supplied	
Query Syntax	STATus:QUEStionable:ENABle?	
Query Parameters	None	
Query Response	<NRf> = Numeric ASCII value from 0 to 32767	
Description	The Status Questionable Enable command sets the bits in the Questionable Status Enable Register. This command is provided only to comply with the SCPI standard. The Status Questionable Enable query reports the contents of the Questionable Status Enable Register. The VM3608A/3616A does not alter the bit settings of this register and will report the last programmed value.	
Examples	**Command / Query**	**Response *(Description)***
	STAT:QUES:ENAB 64	
	STAT:QUES:ENAB?	64
Related Commands	None	

STATus:QUEStionable:EVENt

Purpose	Queries the Questionable Status Event Register.	
Type	Required SCPI command	
Command Syntax	None - Query Only	
Command Parameters	N/A	
***RST Value**	N/A	
Query Syntax	STATus:QUEStionable [:EVENt]?	
Query Parameters	None	
Query Response	0	
Description	The Questionable Status Event Register is provided for SCPI compliance only. The VM3608A/3616A does not alter the bits in this register and queries always report a 0.	
Examples	**Command / Query**	**Response *(Description)***
	STAT:QUES?	0
Related Commands	None	

SYSTem:ERRor?

Purpose	Queries the Error Queue	
Type	Required SCPI command	
Command Syntax	None – Query Only	
Command Parameters	N/A	
***RST Value**	N/A	
Query Syntax	SYSTem:ERRor?	
Query Parameters	None	
Query Response	ASCII string	
Description	The System Error query is used to retrieve error messages from the error queue. The error queue will maintain the two error messages. If additional errors occur, the queue will overflow and the second and subsequent error messages will be lost. In the case of an overflow, an overflow message will replace the second error message. See the SCPI standard Volume 2: Command Reference for details on errors and reporting them.	
Examples	**Command / Query**	**Response *(Description)***
	SYST:ERR?	-350, "Queue overflow"
Related Commands	None	

SYSTem:VERSion?

Purpose	Queries the SCPI version number for which the VM3608A/3616A complies	
Type	Required SCPI command	
Command Syntax	None - Query Only	
Command Parameters	N/A	
***RST Value**	N/A	
Query Syntax	SYSTem:VERSion?	
Query Parameters	None	
Query Response	Numeric ASCII value	
Description	The System Version query reports the version of the SCPI standard for which the VM3608A/3616A complies.	
Examples	**Command / Query**	**Response *(Description)***
	SYST:VERS?	1994.0
Related Commands	None	

INDEX