VM2608/VM2616

ANALOG-TO-DIGITAL CONVERTER

USER'S MANUAL

P/N: 82-0023-000
Released February 26, 2007

VXI Technology, Inc.

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

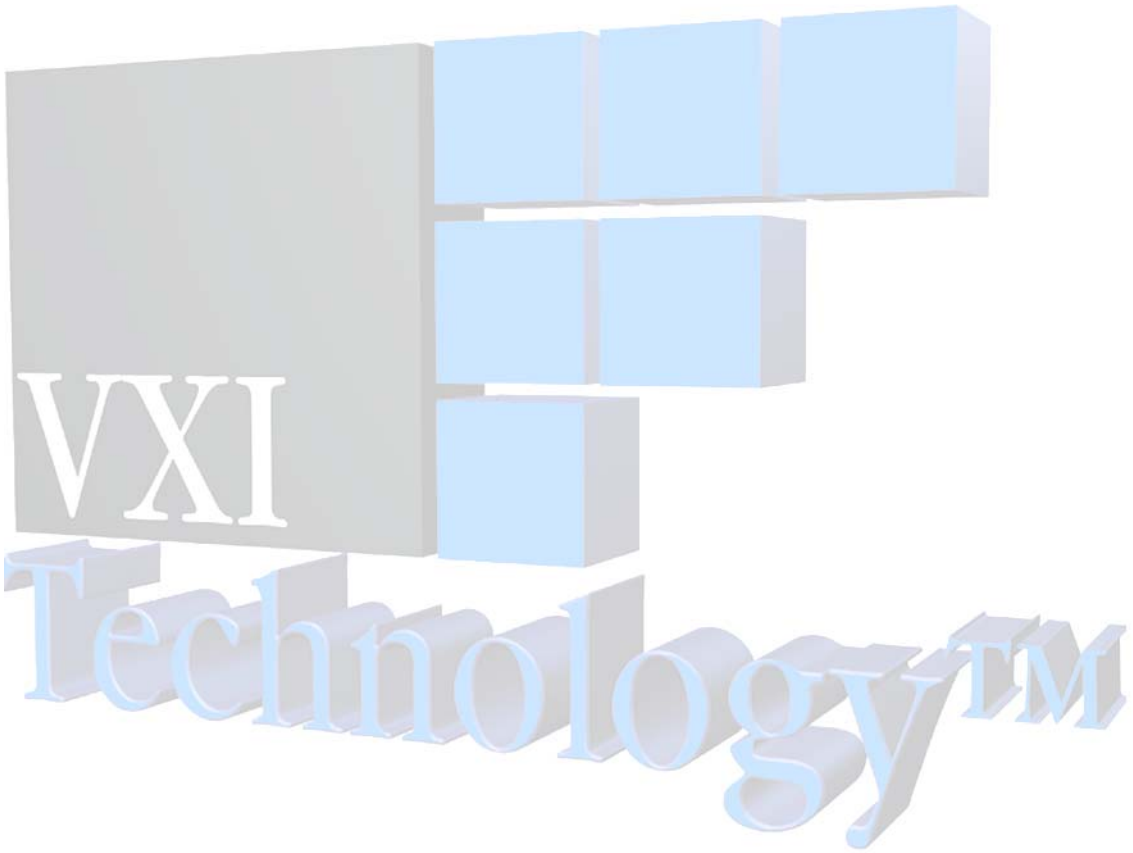

VXI
Technology™

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 VXI Technology 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.

VXI Technology, Inc. shall not be liable for injury to property other than the goods themselves. Other than the limited warranty stated above, VXI Technology, Inc. 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 ISO/IEC Guide 22 and EN 45014

MANUFACTURER'S NAME	VXI Technology, Inc.
MANUFACTURER'S ADDRESS	2031 Main Street Irvine, California 92614-6509
PRODUCT NAME	Analog-to-Digital Converter
MODEL NUMBER(S)	VM2608 & VM2616
PRODUCT OPTIONS	All
PRODUCT CONFIGURATIONS	All

VXI Technology, Inc. declares that the aforementioned product conforms to the requirements of the Low Voltage Directive 73/23/EEC *and the* EMC Directive 89/366/EEC *(inclusive* 93/68/EEC*) and carries the "CE" mark accordingly. The product has been designed and manufactured according to the following specifications:*

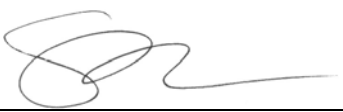
SAFETY	EN61010 (2001)
EMC	EN61326 (1997 w/A1:98) Class A CISPR 22 (1997) Class A VCCI (April 2000) Class A ICES-003 Class A (ANSI C63.4 1992) AS/NZS 3548 (w/A1 & A2:97) Class A FCC Part 15 Subpart B Class A EN 61010-1:2001

The product was installed into a C-size VXI mainframe chassis and tested in a typical configuration.

I hereby declare that the aforementioned product has been designed to be in compliance with the relevant sections of the specifications listed above as well as complying with all essential requirements of the Low Voltage Directive.

February 2007

CE



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 you suspect there is any damage to this product. ***Product should be inspected or serviced only by qualified personnel.***

Improper Use

The operator of this instrument is advised that if equipment is used in a manner not specified in this manual, the protection provided by this equipment be may be impaired.

SUPPORT RESOURCES

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

VXI Technology
World Headquarters

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

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

VXI Technology
Cleveland Instrument Division

5425 Warner Road
Suite 13
Valley View, OH 44125

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

VXI Technology
Lake Stevens Instrument Division

VXI Technology, Inc.
1924 - 203 Bickford
Snohomish, WA 98290

Phone: (425) 212-2285
Fax: (425) 212-2289

Technical Support

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

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

SECTION 1

INTRODUCTION

OVERVIEW

The VM2608/2616 instrumentation module provides 8/16 independent channels of an analog to digital converter (ADC), with 16 bits of resolution and a sample rate of up to 100 kSamples/s (kSa/s). Each channel consists of an independent ADC combined with its own instrumentation amplifier providing a true differential input. The command set conforms to the SCPI standard for consistency and ease of programming

Three VM2608/2616s installed in a single-wide C-size module creates 48 independent A/D channels. The VM2608/2616 may also be combined with any of the other members of the VMIP™ (*VXI Modular Instrumentation Platform*) 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 VM2608/2616 with two other instrument functions in a single-wide, C-size VXIbus module.

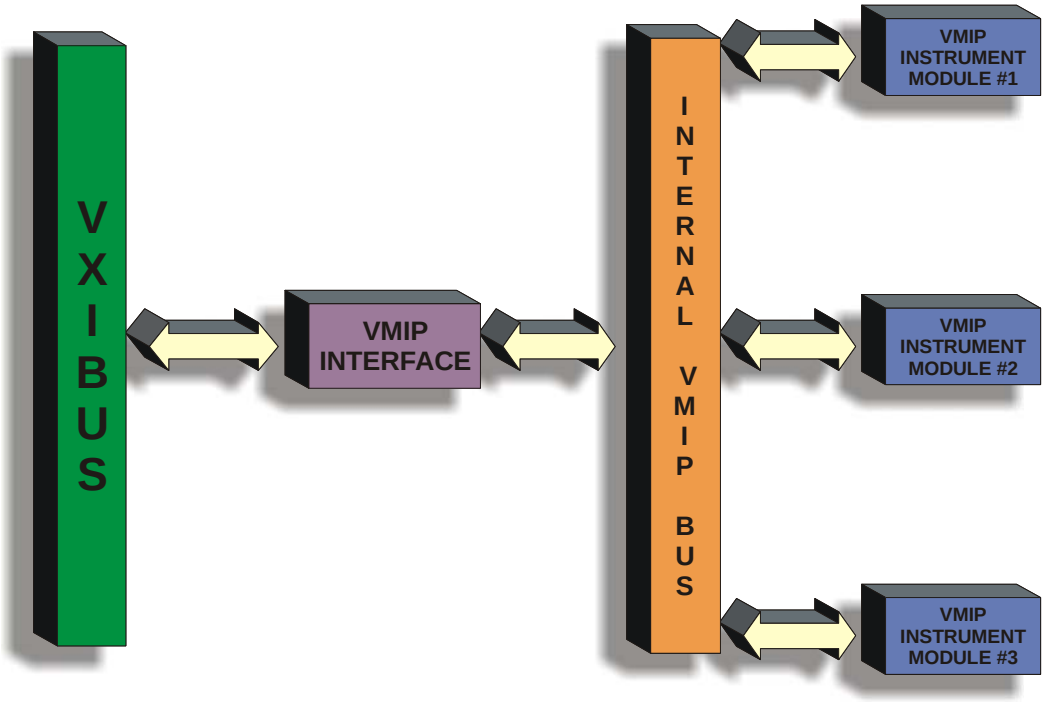
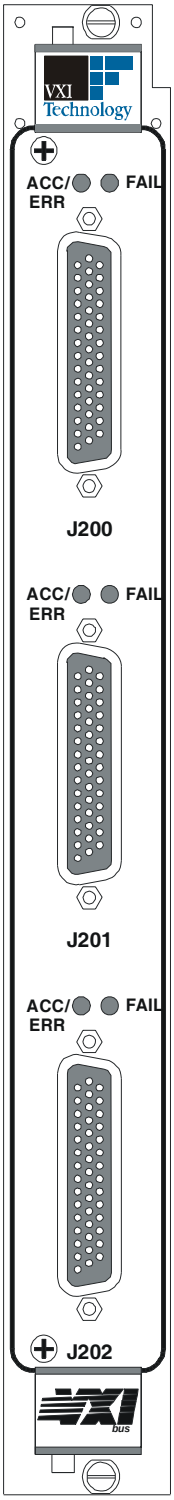


FIGURE 1-1: VMIP™ PLATFORM



DESCRIPTION

The VM2608/2616 instrumentation module provides 8 or 16 independent channels of an analog to digital converter (ADC) with 16 bits of resolution and a sample rate of up to 100 kSa/s. Each channel consists of an independent ADC combined with its own instrumentation amplifier providing a true differential input.

Each group of four channels has a separate bank of memory that may be allocated to one, two, three, or four converters. This allows for the most efficient use of the available memory. The data may also be directly routed to the VXIbus for direct data collection.

The input data is acquired when a conversion trigger event occurs from one of the four following methods:

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 ADCs will convert when a word serial command is received by the instrument.
4) Trigger from one of the 8 or 16 input channels: One of the input channels can be selected to trigger the board.

The trigger event causes all channels to convert simultaneously and pre- and post-triggering is available for added flexibility.

Since three VM2608/2616s can be accommodated on a single C-size VXIbus card, three groups of 8 or 16 channels each can all be operating and triggering independently. For example, one VM2608/2616 can be acquiring data while another is waiting for a trigger.

FIGURE 1-2: FRONT PANEL LAYOUT

PROGRAMMING & DATA ACCESS

Word Serial Message-based Data Access: In this mode, the converted data and all other functions are accessed via the VXI message-based interface. Commands are sent to query the converted values as well as to initiate functions, such as triggering a conversion or querying the calibration constants of each channel. To ease programming, SCPI command sets are used and VXIplug&play drivers are provided.

Pseudo-Register Data Access: In this mode, the data is accessed as in any register-based VXI module, but is different in that the local microprocessor performs additional functions before passing data to the host controller. For example, the local VMIP microprocessor performs the necessary math to provide calibrated data. After a conversion has been made, the microprocessor reads all ADCs and applies the necessary math to obtain accurate data. The data is then loaded into the user definable VXI registers for access.

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.

VM2608/VM2616 SPECIFICATIONS

GENERAL SPECIFICATIONS

RESOLUTION

16 bits, 15 bits monotonic

SIGNAL INPUT TYPE

differential

SIGNAL INPUT COUPLING

dc

SIGNAL INPUT IMPEDANCE

> 10 MΩ	0.1 V, 1.0 V, 10 V range
400 kΩ Differential	0.4 V, 4.0 V, 40 V range
200 kΩ Single Ended ±1%	0.4 V, 4.0 V, 40 V range

ACCURACY

Input Range	Common Mode Input Range	Resolution	% Reading + % Range	Temp Coefficient (per ºC outside of nominal)	Bandwidth[1] (-6 dB)
±0.1 V	±10 V	3.05 µV	±0.06 ± 0.01	±0.002 ± 0.0020	dc to >10 kHz
±0.4 V	±40 V	12.2 µV	±0.07 ± 0.01	±0.003 ± 0.0025	dc to >10 kHz
±1.0 V	±10 V	30.5 µV	±0.05 ± 0.01	±0.002 ± 0.0020	dc to >20 kHz
±4.0 V	±40 V	122 µV	±0.06 ± 0.01	±0.003 ± 0.0025	dc to >20 kHz
±10.0 V	±10 V	305 µV	±0.05 ± 0.01	±0.002 ± 0.0020	dc to >20 kHz
±40.0 V	±40 V	1.22 mV	±0.05 ± 0.01	±0.003 ± 0.0025	dc to >20 kHz

[1] *Bandwidth specifications are measured with full-scale range sine wave input*
1-year specifications with 1 hr warm-up, averaged over 10,000 samples
Specifications are valid at 23 ºC ± 5 ºC

REFERENCE OSCILLATOR

1 MHz derived from the VXI 10 MHz ECL clock

SAMPLING CLOCK

reference oscillator divided by 10 to 16,777,215

SAMPLE RATE

100 kSamples/s to 9.5367 samples per second

SAMPLE MEMORY	STANDARD MEMORY	512 kWORD OPTION
One active channel	128 kWords (131,071)	512 kWords (524,287)
Two active channel	64 kWords (64,535)	256 kWords (262,143)
Three active channel	42.6 kWords (43,689)	170.6 kWords (174,761)
Four active channel	32 kWords (32,767)	128 kWords (131,071)

POWER REQUIREMENTS

VM2608/2616-1	1.42 A @ +5 V, 0.12 A @-5.2 V, 0.18 A @+24 V, 0.10 A @ -24 V
VM2608/2616-2	2.10 A @ +5 V, 0.19 A @-5.2 V, 0.36 A @+24 V, 0.20 A @ -24 V
VM2608/2616-3	2.78 A @ +5 V, 0.26 A @-5.2 V, 0.54 A @+24 V, 0.30 A @ -24 V

SECTION 2

PREPARATION FOR USE

INSTALLATION

When the VM2608/2616 is unpacked from its shipping carton, the contents should include the following items:

(1) VM2608/2616 VXIbus module
(1) VM2608/2616 ADC Module User's (this manual)

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

Once the VM2608/2616 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 zero. The chassis should be checked to ensure that it is capable of providing adequate power and cooling for the VM2608/2616. Once the chassis is found adequate, the VM2608/2616's logical address and the backplane jumpers of the chassis should be configured before the VM2608/2616'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.

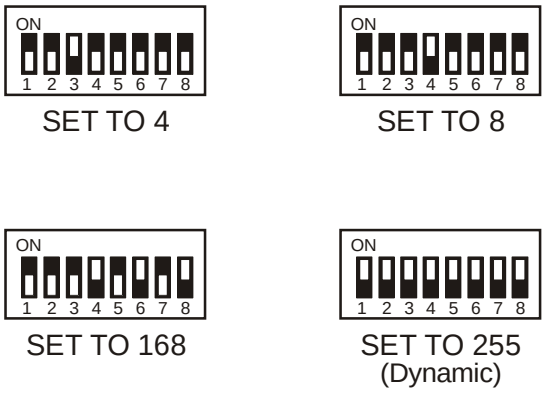
It should be noted that if the chassis cannot provide adequate power to the module, the instrument may not perform to specification or possibly not operate at all. In addition, if adequate cooling is not provided, the reliability of the instrument will be jeopardized and permanent damage may occur. Damage 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 manual for further details on setting the backplane jumpers.

SETTING THE LOGICAL ADDRESS

The logical address of the VM2608/2616 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 logic 1; switches pushed away from the ON legend will signify 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.

FRONT PANEL INTERFACE WIRING

The VM2608/2616's serial interface is made available on the front panel of the instrument. The 8/16-channel version (VM2608/2616-1) will have a J201 that contains all signals for this instrument. The 16/32-channel version (VM2608/2616-2) will have J201 and J202 provided, while the 24/48-channel version (VM2608/2616-3) will have J200, J201, and J202. The wiring for each of these connectors is identical and since each group of four 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 MODEL VM2608/2616 ADC

SIGNAL	PIN	SIGNAL	PIN	SIGNAL	PIN	SIGNAL	PIN
INPUT1+	1	INPUT13-	12	GND	23	INPUT6+	34
INPUT1-	2	GND	13	INPUT11+	24	INPUT6-	35
GND	3	TRIGIN*	14	INPUT11-	25	INPUT9+	36
INPUT4+	4	NC	15	INPUT14+	26	INPUT9-	37
INPUT4-	5	INPUT2+	16	INPUT14-	27	GND	38
INPUT7+	6	INPUT2-	17	GND	28	INPUT12+	39
INPUT7-	7	GND	18	INPUT16+	29	INPUT12-	40
GND	8	INPUT5+	19	INPUT16-	30	INPUT15+	41
INPUT10+	9	INPUT5-	20	INPUT3+	31	INPUT15-	42
INPUT10-	10	INPUT8+	21	INPUT3-	32	GND	43
INPUT13+	11	INPUT8-	22	GND	33	NC	44

The input connector for the VM2608/2616 ADC boards is a 44-pin female high-density D-sub type. Connections listed are for the model VM2616, 16-channel ADC board. A solder pot type mating connector is provided with each unit. Contact the factory for more information on connectors. The pin locations for J200, J201, and J202 are shown in Figure 2-2.

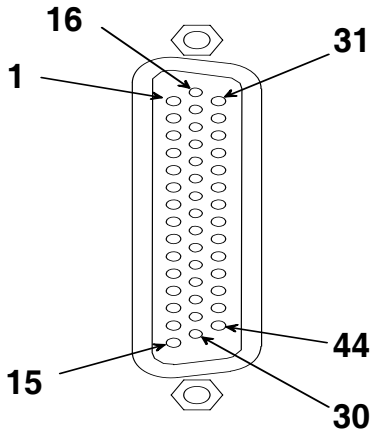


FIGURE 2-2: J200, J201 AND J202 PIN LOCATIONS

SECTION 3

PROGRAMMING

INTRODUCTION

The VM2608/2616 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 uses 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, end each branch with a semicolon. For example, **SLOPe** and **SOURce** are both branches off the **TRIGger:** command trunk and can be combined as follows:

TRIGger:SLOPe <slope>;SOURce <source>

The above command is the same as the these two Commands

TRIGger:SLOPe <slope>
TRIGger:SOURce <source>

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:SLOPe <slope>
trigger:slope <slope>
TRIGGER:SLOPE <slope>
TRIG:SLOPe <slope>
TRIGger:SLOP <slope>
TRIG:SLOP <slope>
trig:slop <slope>

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:

trigg:slop <slope> incorrect syntax - extra "g"

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.

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.

EXAMPLES OF SCPI COMMANDS

ABORt

The Abort command disarms the VM2608/2616 and stops data sampling if active.

ABORt	There are no command parameters.

EXAMPLES

ABORt	*Disarms the VM2608/2616 and stops data sampling if active.*

CALibration:COUNt?

The Calibration Count query returns a number that indicates the number of times the VM2608/2616 has been calibrated. The instrument will increment the count every time the non-volatile memory storing the calibration constants is updated.

CALibration:COUNt?	Where the maximum value for count is 16,777,215 after which it will wrap to zero.

EXAMPLES

CALibration:COUNt?	5 *(Returns a number (5) that indicates the VM2608/2616 has been calibrated five times. There are no query parameters.)*

CALibration:DEFault

The Calibration Default command sets all the calibration gain and offset values to their respective defaults, (i.e., zero).

CALibration:DEFault	There are no command parameters.

EXAMPLES

CAL:SEC:STAT OFF,#16VM2616	*Disabling security.*
CAL1:GAIN 5	*Programming Channel 1's gain.*
CAL1:ZERO -4	*Programming Channel 1's offset*
CALibration:DEFault	*Setting calibration gain -4 offset values of all channels to default.*
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; its effect is immediate. It is important to note that the calibration security must be disabled for the calibration gain command to function.

CALibration <channel>:GAIN <value>	Where <channel> is 1 through 16 or 1 through 8. Where <value> ranges from -128 to +127.

EXAMPLES

CALibration 2:GAIN 75	*Sets the gain value of Channel 2 to 75.*
CALibration 2:GAIN?	75 *(Returns the gain value of Channel 2, which is currently set as 75.)*

CALibration:RESet

The Calibration Reset command resets the calibration values from the non-volatile memory.

CALibration:RESet	There are no command parameters.

EXAMPLES

CALibration:RESet	*Restores the calibration values from the non-volatile memory.*

CALibration:SECure:CODE

The Calibration Secure Code command sets the code required to disable the calibration security. It is important to note that the calibration security state must be disabled in order to change the code string. The default code set by the factory is 'VM2608' for the VM2608 and 'VM2616' for the VM2616.

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

EXAMPLES

CAL:SEC:STAT OFF, #16VM2616	*Disabling security*.
CALibration:SECure:CODE #15OLIVE	*Sets the security code for VM2608/2616 in IEEE 488.2 definite or indefinite length arbitrary block format. The new code is "OLIVE".*
CAL:SEC:STAT ON	*Enabling security*.

CALibration:SECure:STATe

The Calibration Secure State command enables or disables the calibration security. When the 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, the calibration security must be OFF or disabled. In order to disable the security state, the security code must be supplied in a 4-part block format. The four parts are:

1) #
2) A single digit that tells how many digits are in the length
3) Length of the block
4) The actual data (in this case, the characters of the password)

CALibration:SECure:STATe <boolean>,<block>	Where <boolean> is 0 | OFF | 1 | 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 <block> is the parameter that must be present to disable the security, which comprises of four parts as described above.

EXAMPLES

CALibration:SECure:STATe OFF, #16VM2616	*Sets the security OFF so that the calibration constants can be stored in non-volatile memory. The password here is assumed to be "VM2616". Note that the password is case sensitive.*
CALibration:SECure:STATe?	OFF *(Returns the state of the security, which is currently OFF.)*
CALibration:SECure:STATe ON	*Sets the calibration security state ON so that the calibration constants may not be stored in the non-volatile memory.*
CALibration:SECure:STATe?	ON *(Returns the state of the security, which is currently ON).*

CALibration:STORe

The Calibration Store command stores 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; it will generate an error.

CALibration:STORe	The security state should be OFF before using this command. There are no command parameters.

EXAMPLES

CAL:SEC:STAT OFF,#16VM2616	*Disabling security.*
CAL 1:GAIN 25	*Programming Channel 1's gain.*
CAL 1:ZERO -4	*Programming Channels 1's offset.*
CAL 2:GAIN 4	*Programming Channel 2's gain.*
CAL 2:GAIN -1	*Programming Channel 2's offset.*
CALibration:STORe	*Storing the newly programmed calibration constants to the non-volatile memory.*
CAL:SEC:STAT ON	*Enabling security.*

CALibration:ZERO

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

CALibration <channel>:ZERO <value>	Where <channel> is 1 through 8 or 1 through 16. Where <value> ranges from -128 to +127.

EXAMPLES

CALibration 2:ZERO 100	*Sets the calibration offset value for the Channel 2 to 100.*
CALibration 2:ZERO?	100 *(Returns the calibration offset value for Channel 2, which is currently set as 100.)*

FETCh:AVErage?

The Fetch Average query retrieves the average of the data that was collected on the previous run for the specified channel or channels. The data is retrieved based on the starting and ending point specified.

FETCh:AVErage?[<fetch_counts>[,<fetch_address>][,<channel_list>]]

Where <fetch_counts> specifies how many data points are to be retrieved. A <fetch_counts> of 0 specifies data retrieval from <fetch_address> to end of memory.

Where <fetch_address> specifies the starting point for the data retrieval. The value of 0 represents the trigger point, a negative value represents pre-trigger information, and a positive value represents post-trigger information.

Where <channel_list> is the standard channel list format supporting Channels 1 through 8 (VM2608) or 1 through 16 (VM2616). If the channel list is not supplied, all channels are used.

EXAMPLES

FETCh:AVErage? 3, -3, (@1)	1.634528 *(Retrieves the average data value from the data collected on Channel 1, starting at 3 before the trigger point.)*
FETCh:AVErage? 0, 0, (@1)	1.734218 *(Retrieves the average data value from the data set collected for Channel 1, starting at the trigger point and continuing to the end of memory.)*
FETCh:AVErage? (@4)	2.234683 *(Retrieves the average data value from all of the data collected for Channel 4.)*
FETCh:AVErage?	0,41.983456,2.543292… <8 or 16 values> *(Retrieves the average data value for the data collected on all channels, starting from 4 points after the trigger point until the end of memory.*

FETCh:DATA?

The Fetch Data query retrieves the actual data collected on the previous run for the specified channel or channels. The data is retrieved based on the starting and ending point specified.

FETCh:DATA?[<fetch_counts>[, <fetch_address>][, <channel_list>]]

Where <fetch_counts> specifies how many data points are to be retrieved. A <fetch_counts> of 0 specifies data retrieval from <fetch_address> to end of memory.

Where <fetch_address> specifies the starting point for the data retrieval. The value of 0 represents the trigger point, a negative value represents pre-trigger information, and a positive value represents post-trigger information.

Where <channel_list> is the standard channel list format supporting Channels 1 through 8 (VM2608) or 1 through 16 (VM2616). If the channel list is not supplied, all channels are used.

EXAMPLES

FETCh:DATA?3, -3, (@1)	0.543281, 0.954863, 0.982543 *(Retrieves 3 data values from data collected on Channel 1 starting at 3 before the trigger point.)*
FETCh:DATA? 0, 0, (@2)	0.346893, 0.356394, 0.438421… (until end of memory) *(Retrieves data for Channel 2, starting at the trigger point until the end of memory.)*

NOTE You may experience a time out due to lack of data. This will occur when the number of <fetch_counts> exceeds the number of captured data points, or runs over the data points due to being assigned a <fetch_address> that puts the number of <fetch_counts> beyond the number of captured data points. This may be due to the INITiate:DELay command default setting. INITiate:DELay defaults to a value of 1, causing one pre-trigger event to occur.

SWE:POIN 100	*Sets the number of data points to be stored to 100.*
FETCh:DATA 100, 0, (@1)	*Queries Channel 1 for 100 data points starting at the trigger point.*

Since the INITiate function is not defined as IMMediate, or set to a specified value, the default setting of ***INITiate:DELay 1*** is active. The first data point would be at the <fetch_address> of -1, and the last data point would be at the <fetch_address> of 98. The above set of commands would therefore yield a time out when the data point at <fetch_address> 99 (the 100th from 0) was queried and found to have no data.

To correct this fault, either define the INITIATE function, or initiate the ***FETCh:DATA?*** command at the <fetch_address> of -1, or use a smaller <fetch_counts> value.

FETCh:MAXimum?

The Fetch Maximum query specifies that the maximum value out of a group of data collected over the previous run is to be returned. The data set on which the query is to be performed is specified by giving the number of points (fetch count) and a starting point (fetch address). The values are retrieved for all the channels specified in the channel list with multiple channels; the values are separated by commas.

FETCh:MAXimum?[<fetch_counts>[, <fetch_address>][, <channel_list>]]

Where <fetch_counts> specifies how many data points are to be retrieved. A <fetch_counts> of 0 specifies data retrieval from <fetch_address> to end of memory.

Where <fetch_address> specifies the starting point for the data retrieval. The value of 0 represents the trigger point, a negative value represents pre-trigger information, and a positive value represents post-trigger information.

Where <channel_list> is the standard channel list format supporting Channels 1 through 8 (VM2608) or 1 through 16 (VM2616). If the channel list is not supplied, all channels are used.

EXAMPLES

FETCh:MAXimum? 3,256,(@1)	6.250321 (*Retrieves the maximum data value from the data collected on Channel 1, starting at 256 after the trigger point and ending 3 points afterward.)*
FETCh:MAXimum? 0,0,(@2,3)	7.543216, 3.134369 *(Retrieves the maximum value from the data collected on Channels 2 and 3, starting at the trigger point until the end of memory.)*
FETCh:MAXimum? 0,20	6.341396,4.937614… <8 or 16 values> *(Retrieves the maximum value from the data collected on all channels, starting at 20 points past the trigger point until the end of memory.)*
FETCh:MAXimum? 20, -256, (@4:6)	4.937614,3.934564,5.349871…….. *(Retrieves the maximum value from the data collected on Channels 4, 5 and 6, starting at 256 points before the trigger point until 20 points afterward.)*

FETCh:MINimum?

The Fetch Minimum query specifies that the minimum value out of a group of data collected over the previous run is to be returned. The data set on which the query is to be performed is specified by giving the number of points (fetch count) and a starting point (fetch address). The values are retrieved for all the channels specified in the channel list with multiple channels; the values are separated by commas.

FETCh:MINimum?[<fetch_counts>[, <fetch_address>][, <channel_list>]

Where <fetch_counts> specifies how many data points are to be retrieved. A <fetch_counts> of 0 specifies data retrieval from <fetch_address> to end of memory.

Where <fetch_address> specifies the starting point for the data retrieval. The value of 0 represents the trigger point, a negative value represents pre-trigger information, and a positive value represents post-trigger information.

Where <channel_list> is the standard channel list format supporting Channels 1 through 8 (VM2608) or 1 through 16 (VM2616). If the channel list is not supplied, all channels are used.

EXAMPLES

FETCh:MINimum? 512,-512,(@1)	1.250356 *(Retrieves the minimum data value from the data collected on Channel 1, starting at 512 before the trigger point until the trigger point.)*
FETCh:MINimum? 0,1,(@2,3)	1.324695, 2.138794 *(Retrieves the minimum value from the data collected on Channels 2 and 3, starting at one point after the trigger point until the end of the memory.*
FETCh:MINimum? 0,0	1.938764, 2.349864 …<8 or 16 values> *(Retrieves the minimum data value from the data collected on all channels, starting at the trigger point until the end of memory.*
FETCh:MINimum? 0,-256,(@4:8)	1.349867,0.938754,1.385467,2.987654 *(Retrieves the minimum data value from the data collected on Channels 4 through 8, starting at 256 points before the trigger point and continuing until the end of memory.)*

FETCh:NTRansition?

The Fetch N Transition query returns the first negative transition of a group of data collected over the previous run. The data set on which the query is to be performed is specified by giving the number of points (fetch count) and a starting point (fetch address). The values are retrieved for all the channels specified in the channel list with multiple channels; the values are separated by commas.

FETCh:NTRansition?[<fetch_counts>[, <fetch_address>][, <channel_list>]]

Where <fetch_counts> specifies how many data points are to be retrieved. A <fetch_counts> of 0 specifies data retrieval from <fetch_address> to end of memory.

Where <fetch_address> specifies the starting point for the data retrieval. The value of 0 represents the trigger point, a negative value represents pre-trigger information, and a positive value represents post-trigger information.

Where <channel_list> is the standard channel list format supporting Channels 1 through 8 (VM2608) or 1 through 16 (VM2616). If the channel list is not supplied, all channels are used.

EXAMPLES

FETCh:NTRansition? 1024,(@ 16)

-1.389654, 7
(Retrieves the largest negative transition data value from the data collected on Channel 16, starting at the trigger point and continuing for 1024 locations after the trigger point. The largest negative transition value and the address at which the negative transition occurred is returned.)

FETCh:PP?

The Fetch PP query returns the peak-to-peak value out of a group of data collected over the previous run. The data set on which the query is to be performed is specified by giving the number of points (fetch count) and a starting point (fetch address). The values are retrieved for all the channels specified in the channel list with multiple channels; the values are separated by commas.

FETCh:PP?[<fetch_counts>[, <fetch_address>][, <channel_list>]]

Where <fetch_counts> specifies how many data points are to be retrieved. A <fetch_counts> of 0 specifies data retrieval from <fetch_address> to end of memory.

Where <fetch_address> specifies the starting point for the data retrieval. The value of 0 represents the trigger point, a negative value represents pre-trigger information, and a positive value represents post-trigger information.

Where <channel_list> is the standard channel list format supporting Channels 1 through 8 (VM2608) or 1 through 16 (VM2616). If the channel list is not supplied, all channels are used.

EXAMPLES

FETCh:PP? 1024, -512, (@1)	3.654892 (*Retrieves the peak-to-peak values from the data that was collected on Channel 1, starting at 512 before the trigger point and continuing for 1024 locations).*
FETC:PP? 0, -512	3.843961, 4.389656, 3.987632,... <8 or 16 value> *(Retrieves the peak-to-peak values from the data collected on all channels, starting from 512 points before the trigger point and continuing until the end of memory.)*

FETCh:PTRansition?

The Fetch P Transition query returns the first positive transition value out of a group of data collected over the previous run. The data set on which the query is to be performed is specified by giving the number of points (fetch count) and a starting point (fetch address). The values are retrieved for all the channels specified in the channel list with multiple channels; the values are separated by commas.

FETCh:PTRansition?[<fetch_counts>[, <fetch_address][, <channel_list<]

Where <fetch_counts> specifies how many data points are to be retrieved. A <fetch_counts> of 0 specifies data retrieval from <fetch_address> to end of memory.

Where <fetch_address> specifies the starting point for the data retrieval. The value of 0 represents the trigger point, a negative value represents pre-trigger information, and a positive value represents post-trigger information.

Where <channel_list> is the standard channel list format supporting Channels 1 through 8 (VM2608) or 1 through 16 (VM2616). If the channel list is not supplied, all channels are used.

EXAMPLES

FETCh:PTRansition? 32768,-16384,(@16)

0.654892,2.385496
(Retrieves the largest positive transition data value from the data that was collected on Channel 16, starting at 16,384 prior to the trigger point and continuing for 16384 locations after the trigger point. The largest positive transition value and the address at which the positive transition occurred are returned.)

FETCh:TRMS?

The Fetch TRMS query returns the True RMS value out of a group of data collected over the previous run. The data set on which the query is to be performed is specified by giving the number of points (fetch count) and a starting point (fetch address). The values are retrieved for all the channels specified in the channel list with multiple channels; the values are separated by commas.

FETCh:TRMS? [<fetch_counts>[, <fetch_address>][, <channel_list>]]

Where <fetch_counts> specifies how many data points are to be retrieved. A <fetch_counts> of 0 specifies data retrieval from <fetch_address> to end of memory.

Where <fetch_address> specifies the starting point for the data retrieval. The value of 0 represents the trigger point, a negative value represents pre-trigger information, and a positive value represents post-trigger information.

Where <channel_list> is the standard channel list format supporting Channels 1 through 8 (VM2608) or 1 through 16 (VM2616). If the channel list is not supplied, all channels are used.

EXAMPLES

FETCh:TRMS? 10891, -7891, (@12)	3.597634 *(Retrieves the true RMS value from the data that was collected on Channel 12, starting at 7891 locations before the trigger point and continuing for 3000 locations after the trigger point.)*
FETCh:TRMS? 0, 0	0.963481, 3.987654,... <8 or 16 values> *(Retrieves the true RMS value from the data collected on all Channel 4, starting at the trigger point and continuing until the end of memory.)*

INITiate:DELay

The Initiate Delay command arms the VM2608/2616 module, after the selected number of sample points has been taken after the command is received. It is important to note that this command is used to guarantee that the pre-trigger information is valid. In addition, an ***INITiate:IMMediate*** command must follow the INITiate:DELay command to actually arm the device; Initiate Delay only sets up the delay until the device is armed.

INITiate:DELay <sample_points>	Where <sample_points> is 1 to maximum memory (128k or 512k).

EXAMPLES

INITiate:DELay 50	*Arms the VM2608/2616 after the selected number of sample points (50) have been taken after the command is received.*
INITiate:DELay?	50 *(Returns the number of sample points (50) that was configured.)*

INITiate[:IMMediate]

The Initiate Immediate command arms the VM2608/2616 for data sampling, upon receipt of the command.

INITiate[:IMMediate]	There are no command parameters.

EXAMPLES

INITiate[:IMMediate]	*Arms the VM2608/2616 for data sampling.*

OUTPut:TRIGger:SLOPe

The Output Trigger Slope command sets the active slope of the trigger driven onto the TTL trigger bus.

OUTPut:TRIGger:SLOPe <slope>	Where <slope> is either POSitive or NEGative.

EXAMPLES

OUTPut:TRIGger:SLOPe POSitive	*Sets the active slope of the trigger driven onto the TTL trigger bus as POSitive.*
OUTPut:TRIGger:SLOPe?	POS *(Returns the active slope of the trigger driven onto the TTL trigger bus, which is POS.)*
OUTPut:TRIGger:SLOPe NEGative	*Sets the active slope of the trigger driven onto the TTL trigger bus as NEGative.*
OUTPut:TRIGger:SLOPe?	NEG *(Returns the active slope of the trigger driven onto the TTL trigger bus, which is NEG.)*

OUTPut:TTLTrig[:STATe]

The Output TTLTrig State command enables or disables the driving of the trigger signal onto the VXIbus backplane TTL trigger lines. It is important to note that STATE is optional.

OUTPut:TTLTrig[:STATE] <state>	Where <state> is 0 | OFF | 1 | ON. OFF disables the driving of the trigger signal, while ON enables the driving of the trigger signal onto the VXIbus backplane TTLTrigger lines.

EXAMPLES

OUTPut:TTLTrig:STATe ON	*Enables the output trigger to be driven onto the VXIbus TTL trigger bus lines.*
OUTPut:TTLTrig:STATe?	ON *(Returns the state of the output trigger driven onto the VXIbus TTL trigger bus lines, which is currently ON.)*

OUTPut:TRIGger:TTLTrig

The Output Trigger TTLTrig command selects which of the eight VXIbus TTL trigger lines the module will drive when the output is enabled.

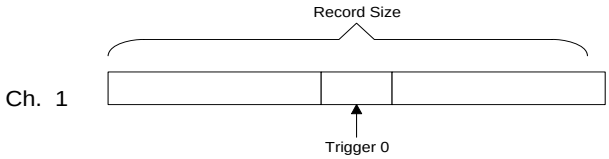
OUTPut:TRIGger:TTLTrig <n>	Where <n> is 0,1,2,3,4,5,6 or 7 VXIbus TTLTrigger lines.

EXAMPLES

OUTPut:TRIGger:TTLTrig 1	*Selects TTL Trigger 1 as output trigger line*
OUTPut:TRIGger:TTLTrig?	1 *(Returns the TTL Trigger line selected for the trigger to be driven onto the backplane TTLTrigger bus.)*

REGister:ADDRess

The Register Address command sets up the record-length location the registers will start to access data. The <address> value is 0 for the trigger point, and is incremented or decremented by 1, to the allowable value determined by the record size.



Once the register address point is determined, data can be accessed in register format over the VXIbus. It is important to note that the *OPC? command should be used, before the data is accessed, to ensure all word serial-commands have been processed.

REGister:ADDRess <address>[, channel_list]	*Where <address> equals 0,-1, or +1* *0 = trigger point* *-1 = trigger point - 1* *+1 = trigger point + 1* Where <channel_list> is the standard channel list format supporting Channels 1 through 8 (VM2608) or 1 through 16 (VM2616). If channel list is not supplied, all the channels are used.

EXAMPLES

REGister:ADDRess 0	*Sets the register address to the trigger point for all channels.*
REGister:ADDRess?	0,0,0,... (8 or 16 values) *(Returns the register address to the trigger point for all channels.)*
REGister:ADDRess -1, (@1:4)	*Sets the register address to the trigger point -1 for Channels 1 through 4.*
REGister:ADDRess? (@1:4)	-1,-1,-1 *(Returns the register address to the trigger point -1 for Channels 1 through 4.)*

ROUTe:CLOSe

The Route Close command enables the channels in the channel list. Enabled channels will record data and will require memory resources. Each group of four channels has its own memory associated with the group. The available memory per group is evenly allocated between the channels that are routed closed. The following channels are also OR'ed together (1, 5, 9, 13), (2, 6, 10, 14), (3, 7, 11, 15), (4, 8, 12, 16). This implies that if Channel 1 and Channel 6 are routed closed, the memory allocated for the first group (Channels 1, 2, 3, and 4), and the second group (Channels 5, 6, 7, and 8) would be allocated to two channels.

ROUTe:CLOSe <channel_list>	Where <channel_list> is the standard channel list format supporting Channels 1 through 8 (VM2608) or 1 through 16 (VM2616). If channel list is not supplied, all the channels are used.

EXAMPLES

ROUTe:CLOSe (@1,2,4)	*Enables Channels 1, 2, and 4 for data recording.*
ROUTe:CLOSe?	1, 2, 4 *(Returns the list of channels that are enabled in the channel list (i.e., 1, 2 and 4).)*

ROUTe:OPEN

The Route Open command disables the channels in the channel list. Disabled channels will not record data and will not use memory resources.

ROUTe:OPEN <channel_list>	Where <channel_list> is the standard channel list format supporting Channels 1 through 8 (VM2608) or 1 through 16 (VM2616). If channel list is not supplied, all the channels are used.

EXAMPLES

ROUTe:OPEN (@1,2,4)	*Disables Channels 1, 2 and 4 from collecting data and using memory resources.*
ROUTe:OPEN?	1,2,4 *Returns the channels that are disabled from collecting data and using memory resources, (i.e., 1, 2, 4, assuming all other channels were previously closed).*
ROUT:CLOS (@1)	*Enabling Channel 1 for data recording.*
ROUTe:OPEN?	2,4 *(Returns the channels that are disabled from collecting data and using memory resources, i.e., 2 and 4 (assuming all other channels were previously closed).)*

SWEep:POINts

The Sweep Points command sets the number of points in a record. The number of points must be *N*, where *N* ranges from two to the maximum memory allowable, as defined in the following table:

Active Channels	Standard Memory	512K Word Option
1	128K words (131,071)	512K words (524,287)
2	64K words (64,535)	256K words (262,143)
3	42.6K words (43,689)	170.6K words (174,761)
4	32K words (32,767)	128K words (131,071)

All channels acquire data simultaneously, so the number of sweep points set would apply to all channels.

SWEep:POINts <points>	Where <points> is the number of samples to be acquired. It ranges from 2 to 128K for 128K on-board memory and 2 to 512K for 512K on-board memory.

EXAMPLES

SWEep:POINts 131072	*Sets the sweep points to all of the available memory for the standard memory configuration or ¼ memory size for 512K memory.*
SWEep:POINts?	131072 *(Returns the number of sweep points in a records, i.e., 128K.)*
SWEep:POINts 32760	*Sets the sweep points to 32K samples.*

SWEep:STEP

The Sweep Step command sets the time interval between samples (time is specified in seconds). This is the sampling rate of the digitizer and is the same for all channels. The sample rate can be set from 100 kHz (0.000010) to 9.5367 Hz (0.1048575).

SWEep:STEP <time> | default	Where <time> specifies the time in seconds from 0.000010 (100 kHz) to 0.1048575 (9.5367 Hz). The default value is 0.000010 (100 kHz).

EXAMPLES

SWEep:STEP 0.0010	*Sets the time interval between samples for all channels to the value of 0.001 seconds.*
SWEep:STEP?	0.0010 *(Returns the time interval between samples, i.e., 0.0010 seconds.)*
SWEep:STEP	*Sets the time interval between samples for all channels to default value of 100 kHz (0.000010 Hz).*

SYNC

The Sync Command allows synchronization of multiple VM2608/2616 modules in a VXIbus chassis, in order to allow multiple modules to simultaneously acquire data. Each card is programmed to trigger off a VXIbus trigger line, and one card is set to generate a sync pulse to that trigger line.

SYNC <mode>	Where <mode> is OUTPut 1 | OUTPut 2, OUTPut 3....OUTPut 7, which is the VXIbus TTL trigger line of the master VM2608/2616 module. INPut 0 | INPut 1.....INPut 7 are the slave VM2608/2616 TTL trigger lines. OFF is used to disable synchronization of multiple VM2608/2616 modules.

EXAMPLES

SYNC OUTPut 0	*Sets TTL trigger line 0 as the master sync output line.*
SYNC OFF	*Disables synchronization of multiple VM2608/2616 modules in the VXIbus chassis.*
SYNC?	OFF *(Reports that synchronization of multiple VM2608/2616 modules is disabled.)*

TRACe:LENGth?

The Trace Length query returns the actual size of data captured during the previous run.

TRACe:LENGth?	There are no query parameters.

EXAMPLES

TRACe:LENGth?	500 *(Returns the actual size of data captured during the previous run, (i.e., 500.))*

TRACe:POINts?

The Trace Points query returns the potential length of the trace that can be captured based upon the current settings.

TRACe:POINts?	There are no query parameters.

EXAMPLES

TRACe:POINts?	32768 *(Returns the potential length of the trace that can be captured (i.e., 32768).)*

TRIGger[:IMMediate]

The Trigger Immediate command is a word serial trigger command that initiates data capture.

TRIGger[:IMMediate]	There are no command parameters.

EXAMPLES

TRIGger[:IMMediate]	*Word serial trigger command to initiate data capture.*

NOTE	This command takes effect only when the module is armed for a trigger. This command performs the same function as *TRG.

TRIGger:LEVel

The Trigger Level command sets the trigger level for the selected trigger source, given that the source is either the front-panel external inputs, or one of the measurement inputs.

TRIGger:LEVel <trigger_level> | default	Where <trigger_level> is a value within the entire input range at the selected gain (i.e., the 1 V range can have a trigger level from -1.0 V to +1.0 V). The default value is 0.0 V.

EXAMPLES

TRIGger:LEVel 1.5	*Sets the trigger level for the selected trigger source as 1.5 V.*
TRIGger:LEVel?	1.5 *(Returns the trigger for the selected trigger source, i.e., 1.5 V.)*
TRIGger:LEVel	*Sets the trigger level to the default value of 0.0 V.*

TRIGger:SLOPe

The Trigger Slope command selects the active edge for triggering the VM2608/2616. Selecting a positive slope will require the trigger input to make a negative to positive transition through the trigger level, while a negative slope requires the trigger input to make a positive to negative transition.

TRIGger:SLOPe <slope>	Where <slope> is either POSitive or NEGative.

EXAMPLES

TRIGger:SLOPe NEGative	*Sets the trigger slope to NEGative. This will require the trigger input to make a transition from positive to negative, through the trigger level.*
TRIGger:SLOPe?	NEG *(Returns that the trigger slope has been set as NEGative.)*

TRIGger:SOURce

The Trigger Source command selects the source of the input trigger used for data capture.

TRIGger:SOURce <source>	Where <source> is CH1 | CH2 | CH3 | CH4 | CH5 | CH6 | CH7 | CH8 | CH9 | CH10 | CH11 | CH12 | CH13 | CH14 | CH15 | CH16 | EXTernal | TTLTrig [0-7] | IMMediate.

EXAMPLES

TRIGger:SOURce IMMediate	*Sets the input trigger source as IMMediate.*
TRIGger:SOURce?	IMM *(Returns the input trigger source that has been set as IMMediate.)*
TRIGger:SOURce CH16	*Sets the input trigger source to Channel 16.*
TRIGger:SOURce?	CH16 *(Returns the source of the input trigger that has been set as Channel 16.)*

VOLTage:RANGe

The Voltage Range command sets the input range for the channels listed in the channel list. It is important to note that the 40 volt range is both the power up condition and the *RST condition.

VOLTage:RANGe <range>, <channel_list>	Where <range> is 0.1 | 0.4 | 1 | 4 | 10 | 40. Where <channel_list> is the standard channel list format supporting Channels 1 through 8 (VM2608) or 1 through 16 (VM2616). If the channel list is not supplied, all the channels are used.

EXAMPLES

VOLTage:RANGe 10, (@1:8)	*Sets the voltage range to ±10 for Channels 1 through 8.*
VOLTage:RANGe? (@1:8)	10 *(Returns the voltage range ±10 for channels 1 through 8.)*

APPLICATION EXAMPLES

This section contains examples of using SCPI command strings for programming the VM2608/2616 module. The code is functional and will contain a brief description of the operation.

Example 1

In this example, the VM2608/2616 sets the calibration gain and calibration offset for the specified channel. The calibration security is turned OFF and the above 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,#16VM2616	Sets the calibration security OFF with the code VTI.
CAL1:GAIN 2	Sets the calibration gain for Channel 1 to 2 volts.
CAL2:ZERO 2	Sets the calibration offset for Channel 2 to 2 volts.
CAL:STOR	Stores the calibration gain and offset values to the non-volatile memory.
CAL:COUN?	Returns the number of times the non-volatile memory has been updated.
CAL:SEC:STAT 1	Disables further stores to the non-volatile memory.

Example 2

In this example, VM2608/2616 sets the sweep points, sweep step, voltage range, and trigger source. It returns the average, data, minimum, maximum, peak to peak, TRMS, and positive or negative transition out of a group of data collected on the previous run. It starts by giving a starting point (fetch address) and the number of points (fetch count) for a selected list of channels.

SWE:POIN 32768	Sets the number of points to 32K.
SWE:STEP 0.00010	Sets the time interval between samples to 10 kHz.
VOLT:RANG 40	Sets the input voltage range to ±40 V.
TRIG:SOUR IMM	Sets the input trigger source to IMMediate.
OUTP:TTLT:STAT ON	Enabling trig output signal.
OUTP:TRIG:TTLT 1	Routing output trigger to TTLT1.
OUTP:TRIG:SLOP NEG	Configuring the triggering edge of the output trigger as NEGative.
ROUT:OPEN (@9:16)	Disables Channels 9 through 16 (VM2616) for acquiring data and utilizing memory resources.
ROUT:CLOSE (@1:8)	Enables Channels 1 through 8 for data capture.
INIT:IMM	Arms the VM2608/2616 module immediately.
TRIG:IMM	Causes a word serial trigger event to occur to initiate data capture.
TRAC:LENG?	32768 Returns the actual length of data captured, which is 32K.
FETC:AVE? (@1:8)	15.107537, 13.478260, 12.37925, 14,654972, 12.897630, 11.391530, 14.394923, 16.789462 Returns the average data value for channels 1 through 8.
FETC:DATA? 8, 0 (@1:8)	0.026730,1.543210,10.543210,6.453210, 13.478230,9.999820,0.86452,8.143210, 15.107537,13.478260,12.37925,14,654972, 12.897630,11.391530,14.394923,16.789462… (+ 48 more values) (Retrieves 8 data values for Channels 1 through 8 starting at the trigger point. 8 values for Channel 1 are returned, then 8 values for Channel 2, etc.)

FETC:MAX? (@1:8)	13.478560,16.789462,14.969493,18.76543, 3.497621,19.234100,8.976293,18.976234 (Returns the maximum data value collected on Channels 1 through 8.)
FETC:MIN? (@1:8)	-0.026730, -11.391530, -6.397824, -12.398726, -1.395243, -12.693521, -3.422216, -8.912342 (Returns the maximum data value collected on Channels 1 through 8.)
FETC:NTR? (@1:8)	0.095698,32,-1.983784,128, 3.672654,265,-6.785469,672 4.543264,195,-8.643264,548 1.943948,230,-3.454545,455 (Returns the first negative transition data value from all data collection Channels 1 through 8, starting at the trigger point and continuing to the end of data collected.)
FETC:PTR? (@1:8)	0.015898,783,8.983784,111, 8.526654,452,6.739769,656 1.452479,894,5.454545,100 2.758456,727,3.456789,111 (Returns the first positive transition data value from all data collection Channels 1 through 8, starting at the trigger point and continuing to the end of data collected.)
FETC:PP? (@1:8)	13.451500,5.397932,8.571669,6.366704 2.152378,6.540579 5.554077,10.063892 (Returns the peak-to-peak value from all data collected on Channels 1 through 8, starting at the trigger point and continuing to the end of data collected.)
FETC:TRMS? (@1:8)	4.653781, 2249768, 1.707895, 8.651232 3.335965, 7.565465, 6.333989, 5.123456 (Returns the true RMS value from all data collected on Channels 1 through 8, starting at the trigger point and continuing to the end of the data collected.)

REGISTER ACCESS EXAMPLES

The VM2608/2616 module provides pseudo register access for fast data transfers. The register map is shown in Table 3-1.

In order to read a voltage for a particular channel the following steps must be followed:

a) Read the register value at the offset corresponding to the channel.
b) Convert the register representation to its equivalent voltage by applying the following formula:

voltage value = ((double) register value) * voltage range / 32768.0

where 'voltage range' is the current range that has been programmed for the channel.

For example, if the register corresponding to Channel 1 is read at offset 0x20 and the value read is 16384, then with Channel 1 operating in the 1 Volt Range, the voltage for Channel 1 is

(16384.0 * 1.0) / 32768.0 = 0.5 volts.

The word serial command REGister:ADDRess command sets up where in the record length, data will start to be accessed by the registers. The <address> value is 0 for the trigger point, and is incremented or decremented by 1 to the allowable value determined by the record size. Once the register address point is determined, data can be accessed in register format over the VXIbus.

For example, if REGister:ADDRess 0,(@1) is issued before Channel 1's register is read, then when the register for Channel 1 is read, the data returned would be the Trigger Point Data for Channel 1.

The VM2608/2616 A/D Module supports access to the 8/16 channels via the Device Dependent Registers of VXI bus interface. The following table shows A16 Memory and the VM2608/2616 Data Port Map.

TABLE 3-1: A16 MEMORY MAP

3E	Channel 16 (VM2616)
3C	Channel 15 (VM2616)
3A	Channel 14 (VM2616)
38	Channel 13 (VM2616)
36	Channel 12 (VM2616)
34	Channel 11 (VM2616)
32	Channel 10 (VM2616)
30	Channel 9 (VM2616)
2E	Channel 8
2C	Channel 7
2A	Channel 6
28	Channel 5
26	Channel 4
24	Channel 3
22	Channel 2
20	Channel 1
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

VXIPLUG&PLAY DRIVER EXAMPLES

/***

Function: vtvm2616_setup_and_fetch_data
Formal Parameter: ViSession instrHndl

A unique handle to the instrument.

ViInt16 enb_dis_output_trigger

This parameter is used to enable or disable the output trigger signal onto the VXIbus backplane TTL trigger lines." On" enables the drivers, while "Off" disables them.

Valid range:

---

vtvm2616_TRIGGER_STATE_OFF or
vtvm2616_TRIGGER_STATE_ON

ViInt16 output_trigger_selection

This parameter selects which VXIbus TTL trigger line the module will drive when the output is enabled.

Valid range:

---

vtvm2616_TTLTRIG0
vtvm2616_TTLTRIG1
vtvm2616_TTLTRIG2
vtvm2616_TTLTRIG3
vtvm2616_TTLTRIG4
vtvm2616_TTLTRIG5
vtvm2616_TTLTRIG6
vtvm2616_TTLTRIG7

ViInt16 output_trigger_slope

This parameter is used to set the active slope of the trigger driven onto the TTL trigger bus.

Valid range:

---

vtvm2616_TRIGGER_SLOPE_NEG
or
vtvm2616_TRIGGER_SLOPE_POS

ViUInt32 sweep_points

This parameter is used to set the number of points in a record.

Valid range:

---

vtvm2616_SWEEP_POINTS_MIN(2) to
vtvm2616_SWEEP_POINTS_MAX(524288)

ViReal64 sweep_time

This parameter is used to set the time interval between samples.

Valid range:

---

vtvm2616_SWEEP_TIME_MIN(0.000010) to
vtvm2616_SWEEP_TIME_MAX(0.1048575)

ViInt16 voltage_range_selection

This parameter is used to set the input voltage range for the channels described by the channel list. The 40.0-volt range is power up condition and reset condition.

Valid range:

---

vtvm2616_VOLTAGE_RANGE1 (0.1v)
vtvm2616_VOLTAGE_RANGE2 (0.4v)
vtvm2616_VOLTAGE_RANGE3 (1v)
vtvm2616_VOLTAGE_RANGE4 (4v)
vtvm2616_VOLTAGE_RANGE5 (10v)
vtvm2616_VOLTAGE_RANGE6 (40v)

ViInt16 no_of_channels

Specify the number of channels in the channel list array.

Valid range:

---

vtvm2616_CHANNEL_NUMBER_MIN(1) to
vtvm2616_MAX_CHANNELS_VM2616(16) for VM2616.
vtvm2616_CHANNEL_NUMBER_MIN(1) to
vtvm2616_MAX_CHANNELS_VM2608(8) for VM2608.

ViPInt16 channel_list

This parameter gives the list of channels selected. The valid range for channels is from 1 to 16 for VM2616 and 1 to 8 for VM2608.

ViInt16 fetch_data_type

This parameter is used to select different types of data read as given below.

vtvm2616_READ_AVERAGE_DATA

This retrieves the average out of a group of data collected on the previous run.

vtvm2616_READ_DATA_LIST

This retrieves the data collected on the previous run.

vtvm2616_READ_MAXIMUM_DATA

This retrieves the maximum out of a group of data collected from the previous run.

vtvm2616_READ_MINIMUM_DATA

This retrieves the minimum out of a group of data collected from the previous run.

vtvm2616_READ_NEG_TRANS_DATA

This retrieves negative transition out of a group of data collected from the previous run.

vtvm2616_READ_PP_DATA

This retrieves the peak to peak value out of a group of data collected from the previous run.

vtvm2616_READ_POS_TRANS_DATA

This retrieves positive transition out of a group of data collected from the previous run.

vtvm2616_READ_TRMS_DATA

This retrieves the true RMS value out of a group of data collected from the previous run.

Valid range:

---

vtvm2616_READ_AVERAGE_DATA
vtvm2616_READ_DATA_LIST
vtvm2616_READ_MAXIMUM_DATA
vtvm2616_READ_MINIMUM_DATA
vtvm2616_READ_NEG_TRANS_DATA
vtvm2616_READ_PP_DATA
vtvm2616_READ_POS_TRANS_DATA
vtvm2616_READ_TRMS_DATA

ViInt16 fetch_type

This parameter is used select Default or user mode.

vtvm2616_USER_SEL

In User mode, data is retrieved for selected list of channels by giving a starting point (fetch address) and the number of points the (fetch count).

vtvm2616_DEFAULT_SEL

In Default mode, data is retrieved from the beginning of memory until the end.

Valid range:

---

vtvm2616_DEFAULT_SEL or
vtvm2616_USER_SEL

ViInt32 fetch_address

This parameter specifies the starting point from where the data is to be retrieved. Fetch address can be negative if the user wants to retrieve the pre trigger data, zero, if the user wants to retrieve data at the trigger point or positive if post trigger data is to be retrieved.

ViInt132 fetch_count

This parameter specifies the number of zero implies that the data should be fetched from <fetch_address> to the end of memory.

ViPReal64 data_arry

The is the numeric array that contains the retrieved data. This array contains the retrieved information and each element corresponds to the channel number in the channel_list array at the same offset except.

vtvm2616_READ_DATA_LIST,
vtvm2616_READ_NEG_TRANS_DATA &
vtvm2616_READ_POS_TRANS_DATA

The following is the list of data retrieved versus the type of data requested.

vtvm2616_READ_AVERAGE_DATA

Average no_of_channels of samples.

vtvm2616_READ_DATA_LIST

*samples (no_of_channels * acquired fetch_count)*

The samples are interleaved in the order of channels as specified in channel_list. <fetch_count> of 0 means the entire data for each specified channel. The maximum value for fetch count can be determined by using

vtvm2616_read_data_capture_info *driver function.*

vtvm2616_READ_MAXIMUM_DATA

Maximum no_of_channels of samples.

vtvm2616_READ_MINIMUM_DATA

Minimum no_of_channels of samples.

vtvm2616_READ_NEG_TRANS_DATA

*Amt of -ve (no_of_channels*2) transition. The first 2 elements are for the first channel specified in the channel_list. The second 2 elements for the second channel in the channel_list and so on. The first element of each ordered pair is the Amt of -ve transition and the second element is the address of -ve transition for a channel.*

vtvm2616_READ_PP_DATA

Peak to no_of_channels peak of samples.

vtvm2616_READ_POS_TRANS_DATA

*Amt of +ve no_of_channels*2 transition. The first two elements are for the first channel specified in the channel_list. The second two elements for the second channel in the channel_list and so on. The first element of each ordered pair is the Amt of +ve transition and the second element is the address of +ve transition for a channel.*

vtvm2616_READ_TRMS_DATA

True RMS no_of_channels of samples.

ViPInt32 num_elements

Specifies the number of elements in data_arry.

Return Values: *Returns VI_SUCCESS if successful, else returns error value.*

Description *This function configures output trigger, and sets sweep points, voltage range. This function also retrieves the AVERAGE, DATA or MAXIMUM or MINIMUM or Peak to Peak or TRMS or +ve/-ve transition out of a group of data collected on the previous run by giving a starting point the (fetch address),the number of points (fetch count) for the selected list of channels. The input trigger is set to Immediate.*

**********/

```
ViStatus _VI_FUNC vtvm2616_setup_and_fetch_data (ViSession      instrHndl,

        ViInt16         enb_dis_output_trigger,
        ViInt16         output_trigger_selection,
        ViInt16         output_trigger_slope,
        ViUInt32        sweep_points,
        ViReal64        sweep_time,
        ViInt16         voltage_range_selection,
        ViInt16         no_of_channels,
        ViInt16         channel_list[],
        ViInt16         fetch_data_type,
        ViInt16         fetch_type,
        ViInt32         fetch_address,
        ViInt32         fetch_count,
        ViReal64        data_arry[],
        ViInt32         *num_elements)

{
    ViStatus status = VI_NULL;

            /* Configure input trigger as Immediate */

        status = vtvm2616_conf_input_trigger(instrHndl,

        vtvm2616_TRIGGER_SOURCE_IMM, 0.0,
        vtvm2616_TRIGGER_SLOPE_POS); if (status < VI_SUCCESS) return
        vtvm2616_CONF_INPUT_TRIGGER_ERROR;

            /* Configure output trigger */

        status = vtvm2616_conf_output_trigger(instrHndl,
            enb_dis_output_trigger, output_trigger_selection, output_trigger_slope);

        if (status < VI_SUCCESS) return
            vtvm2616_CONF_OUTPUT_TRIGGER_ERROR;

            /* Sets the number of points in a record and the time interval between
            samples.*/

        status = vtvm2616_conf_sweep (instrHndl,sweep_points, sweep_time);

        if (status < VI_SUCCESS) return
            vtvm2616_CONF_SWEEP_ERROR;

            /* Sets the input range for the channels described by the channel list */

        status = vtvm2616_set_voltage_range(instrHndl,voltage_range_selection,
            no_of_channels,channel_list);

        if (status < VI_SUCCESS)return
            vtvm2616_SET_VOLTAGE_RANGE_ERROR;
```

```
                    /* Enable all the channels selected */

                status = vtvm2616_enb_or_dis_channels(instrHndl,
                    vtvm2616_ENABLE_CHANNEL,  no_of_channels,  channel_list);

                if (status < VI_SUCCESS) return
                    vtvm2616_ENB_OR_DIS_CHANNELS_ERROR;

                    /* Arm the VM2608/2616 */

                status = vtvm2616_arm_or_disarm(instrHndl,
                    vtvm2616_INITIATE_IMM, vtvm2616_SAMPLE_POINTS_MIN);

                if (status < VI_SUCCESS)return
                    vtvm2616_ARMING_OR_DISARMING_ERROR;

                    /* Send Software Trigger (*TRG) command */

                status = vtvm2616_trigger_event(instrHndl);

                if (status < VI_SUCCESS) return
                    vtvm2616_TRIGGER_EVENT_ERROR;

                    /* Wait for the data capture to take place */

                Delay(5);

                status = vtvm2616_read_data(instrHndl, no_of_channels, channel_list,
                    fetch_data_type, fetch_type, fetch_address,   fetch_count, data_arry,
                    num_elements);

                if (status < VI_SUCCESS)return
                    vtvm2616_READ_DATA_ERROR;

                    /* Disable all the channels selected */

                status =  vtvm2616_enb_or_dis_channels(instrHndl,
                    vtvm2616_DISABLE_CHANNEL, no_of_channels, channel_list);

                if (status < VI_SUCCESS) return
                    vtvm2616_ENB_OR_DIS_CHANNELS_ERROR;

                    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 VM2608/2616 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 *RST 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 VM2608/2616 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 *RST 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	*RST 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.		
*RST	Resets the module to a known state.		N/A
*SRE	Set the Service Request Enable Register.		
*STB?	Query the Status Byte Register.		
*TRG	Causes a trigger event to occur.		N/A
*TST?	Starts and reports a self-test procedure.		N/A
*WAI	Halts execution of commands and queries.	X	

TABLE 4-2: INSTRUMENT SPECIFIC SCPI COMMANDS

Command	Description	*RST	*RST Value
ABORt	Disarms the VM2608/2616 and stops data sampling.		N/A
CALibration:COUNt?	Returns the number of times the VM2608/2616 has been calibrated.		N/A
CALibration:DEFault	Set calibration values to default values.		N/A
CALibration:GAIN	Sets the calibration gain value.		N/A
CALibration:RESet	Resets calibration information from the non-volatile memory.		N/A
CALibration:SECure:CODE	Changes the calibration security code.		N/A
CALibration:SECure:STATe	Enables or disables the Calibration Security for non-volatile memory operations.	X	1
CALibration:STORe	Stores the current calibration constants into the non-volatile memory.		N/A
CALibration:ZERO	Sets the selected channel to zero value.		N/A
FETCh:AVErage?	Retrieves the average value out of a group of data collected during the previous run.		N/A
FETCh:DATA?	Retrieves the data collected during the previous run.		N/A
FETCh:MAXimum?	Retrieves the maximum value out of a group of data collected during the previous run.		N/A
FETCh:MINimum?	Retrieves the minimum value out of a group of data collected during the previous run.		N/A
FETCh:NTRansition?	Retrieves the largest negative transition out of a group of data collected during the previous run		N/A
FETCh:PP?	Retrieves the peak to peak value out of a group of data collected during the previous run.		N/A
FETCh:PTRansition?	Retrieves the largest positive transition out of a group of data collected during the previous run.		N/A
FETCh:TRMS?	Retrieves the true RMS value out of a group of data collected during the previous run.		N/A
FORMat	Sets the calibration constant output format.	X	ASCII
INITiate:DELay	Arms the VM2608/2616 after the selected number of sample points have been taken after the command is received.	X	1
INITiate[:IMMediate]	Arms the VM2608/2616 upon receipt of the command.		N/A
OUTPut:TRIGger:SLOPe	Sets the active slope of the trigger driven onto the TTL trigger bus	X	NEG
OUTPut:TRIGger:TTLTrig	Selects which VXIbus TTL trigger line the module will drive when the output is enabled.	X	TTLT0
OUTPut:TTLTrig[:STATe]	Enables or disables driving the trigger signal onto the VXIbus backplane TTL trigger lines.	X	0
REGister:ADDRess	Determines where in the record length data will start to be accessed by the registers.		Channel 1
ROUTe:CLOSe	Enables the channels in the channel list.	X	All closed
ROUTe:OPEN	Disables the channels in the channel list.	X	All closed
SWEep:POINts	Sets the number of points in a record.	X	32768 or 131072
SWEep:STEP	Sets the time interval between samples.	X	0.000010 s

Instrument Specific SCPI Commands (Cont.)

Command	Description	*RST	*RST Value
SYNC	Allows synchronization between multiple VM2608/2616 modules	X	OFF
TRACe:LENGth?	Returns actual size of the data captured during the previous run		N/A
TRACe:POINts?	Queries potential length that can be captured based on current settings		N/A
TRIGger[:IMMediate]	Word serial trigger command to initiate data capture.		N/A
TRIGger:LEVel	Sets the trigger level for the selected trigger source	X	0.0V
TRIGger:SLOPe	Selects the active edge for triggering the VM2608/2616	X	NEG
TRIGger:SOURce	Selects the source for triggering the VM2608/2616	X	CH1
VOLTage:RANGe	Sets the input range for the channels listed by the channel list	X	All channels to 40.0 V

TABLE 4-3: SCPI REQUIRED COMMANDS

Command	Description	*RST	*RST 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 which version of the SCPI standard the module complies with		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 Event Status Enable Register.
Type	IEEE 488.2 Common Command
Command Syntax	*ESE <mask>
Command Parameters	<mask> = Numeric ASCII value from 0 to 255
***RST Value**	N/A
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/IEEE 488.2-1987 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, an interrupt is generated. See the ESR? command for details regarding the individual bits. The ESE register layout is as follows: Bit 0 - Operation Complete Bit 1 - Request Control (not used in the VM2608/2616) Bit 2 - Query Error Bit 3 - Device Dependent Error (not used in the VM2608/2616) Bit 4 - Execution Error Bit 5 - Command Error Bit 6 - User Request (not used in the VM2608/2616) 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 as follows: Bit 0 - Operation Complete Bit 1 - Request Control (not used in the VM2608/2616, always 0) Bit 2 - Query Error Bit 3 - Device Dependent Error (not used in the VM2608/2616, always 0) Bit 4 - Execution Error Bit 5 - Command Error Bit 6 - User Request (not used in the VM2608/2616, always 0) Bit 7 - Power On The Operation Complete bit is set by the VM2608/2616 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** / **Response *(Description)***
	*ESR? / 4
Related Commands	*ESE

*IDN?

Purpose	Query the module 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 VM2608/2616 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.,2616,0,1.1 *(The revision listed here is for reference only; the response will always be the current revision of the instrument.)*
Related Commands	None	

*OPC

Purpose	Set 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 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 listing at the beginning of this section for the *RST parameter values used with this command.	
Examples	**Command / Query**	**Response *(Description)***
	*RST	
Related Commands	None	

*SRE

Purpose	Set 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**	N/A	
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 backplane 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. 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	*STB	

*STB?

Purpose	Query 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 VM2608/2616, 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	TRIGger[:IMMediate]	

*TST?

Purpose	Starts and reports a self-test procedure.	
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 VM2608/2616 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 value from 0 to 143 which has the following meaning: Bit 0 - Channel 1 Failed Bit 1 - Channel 2 Failed Bit 2 - Channel 3 Failed Bit 4 - Channel 4 Failed Bit 5 - Unused Bit 6 - Unused Bit 7 - RAM Test Failed 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 pattern test or an all zeros test.	
Examples	**Command / Query**	**Response *(Description)***
	*TST?	0
Related Commands	None	

*WAI

Purpose	Halts execution of commands and queries.	
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 processing. It provides a way of synchronizing the module with its master.	
Examples	**Command / Query**	**Response *(Description)***
	*WAI	
Related Commands	*OPC	

DEVICE SPECIFIC SCPI COMMANDS

ABORt

Purpose	Disarms the VM2608/2616 and stops data sampling.	
Type	Event	
Command Syntax	ABORt	
Command Parameters	N/A	
***RST Value**	N/A	
Query Syntax	None	
Query Parameters	N/A	
Query Response	N/A	
Description	The Abort command disarms the VM2608/2616 and stops data sampling (if active).	
Examples	**Command / Query**	**Response *(Description)***
	ABOR	
Related Commands	INITiate:DELay INITiate[:IMMediate]	

CALibration:COUNt?

Purpose	Returns the number of times the VM2608/2616 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	
Description	The Calibration Count query returns the number of times the VM2608/2616 has been calibrated. Calibration values are stored in non-volatile. Non-volatile has a limited life. Each store counts as one cycle. The non-volatile has a guarantee minimum of 10k cycles. Calibration count tracks the number of stores. The maximum value for the count is 16,777,215 after which it will wrap to 0.  **Calibration commands should only be executed by qualified personnel. Changing these values incorrectly can cause the instrument to perform improperly**	
Examples	**Command / Query**	**Response *(Description)***
	CAL:COUN?	5
Related Commands	CALibration:STORe	

CALibration:DEFault

Purpose	Set calibration values to default values.
Type	Setting
Command Syntax	CALibration:DEFault
Command Parameters	None
***RST Value**	N/A
Query Syntax	N/A
Query Parameters	N/A
Query Response	N/A
Description	The Calibration Default command sets the calibration gain and calibration zero values to their respective defaults, (i.e., zero). **Calibration commands should only be executed by qualified personnel. Changing these values incorrectly can cause the instrument to perform improperly**

Examples	**Command / Query**	**Response *(Description)***
	CAL:DEF	

Related Commands	CALibration <channel>:GAIN <value> CALibration <channel>:ZERO <value>

CALibration:GAIN

Purpose	Sets the calibration gain value.
Type	Setting
Command Syntax	CALibration <channel>:GAIN <value>
Command Parameters	<channel> = 1 to 16 (VM2616) or 1 to 8 (VM2608) <value> = -128 to +127
***RST Value**	N/A
Query Syntax	CALibration <channel>:GAIN?
Query Parameters	<channel> = 1 to 16 (VM2616) or 1 to 8 (VM2608)
Query Response	<value> = -128 to +127
Description	The Calibration Gain command is used to set the calibration gain value for the specified channel. The query returns the present calibration gain value for the selected channel. The Calibration Gain command will function only when calibration security is disabled, otherwise, an error is generated.  **Calibration commands should only be executed by qualified personnel. Changing these values incorrectly can cause the instrument to perform improperly**

Examples	**Command / Query**	**Response *(Description)***
	CAL1:GAIN 0	*(This sets Channel 1 to a gain value of 0.)*
	CAL1:GAIN?	0 *(Returns the calibration gain value for Channel 1.)*

Related Commands	CALibration:DEFault CALibration:STORe CALibration <channel>:ZERO <value>

CALibration:RESet

Purpose	Resets calibration information from the non-volatile memory.	
Type	Event	
Command Syntax	CALibration:RESet	
Command Parameters	None	
***RST Value**	N/A	
Query Syntax	N/A	
Query Parameters	N/A	
Query Response	N/A	
Description	The Calibration Reset command resets calibration information to the information stored in non-volatile memory.  **Calibration commands should only be executed by qualified personnel. Changing these values incorrectly can cause the instrument to perform improperly**	
Examples	**Command / Query**	**Response *(Description)***
	CAL:RES	*(Resets the calibration values.)*
Related Commands	None	

CALibration:SECure:CODE

Purpose	Changes the calibration security code.
Type	Setting
Command Syntax	CALibration:SECure:CODE <block>
Command Parameters	<block> = IEEE 488.2 definite length arbitrary block
***RST Value**	N/A
Query Syntax	CALibration:SECure:CODE?
Query Parameters	None
Query Response	IEEE 488.2 definite length arbitrary block
Description	The Calibration Source Code command changes the calibration security code. Calibration values stored in non-volatile are protected by a security code. To be effective, the calibration security state must first be disabled. When the module is shipped from the factory, the calibration security code is either VM2608 or VM2616.  **Calibration commands should only be executed by qualified personnel. Changing these values incorrectly can cause the instrument to perform improperly**

Examples	**Command / Query**	**Response *(Description)***
	CAL:SEC:STAT OFF,#16VM2616	*(Disables calibration security so the code can be changed.)*
	CAL:SEC:CODE #15OLIVE	*(Sets the new security code to "*OLIVE*".)*
	CAL:SEC:CODE?	#15OLIVE *(Returns the calibration security code.)*
	CAL:SEC:STAT ON	*(Enables calibration security.)*

Related Commands	CALibration:SECure:STATe

CALibration:SECure:STATe

Purpose	Enables or disables the Calibration Security for non-volatile memory operations.
Type	Setting
Command Syntax	CALibration:SECure:STATe <boolean>[,<block>]
Command Parameters	<boolean> = 0 | 1 | OFF | ON <block> = IEEE 488.2 definite length arbitrary block
***RST Value**	1 or ON
Query Syntax	CALibration:SECure:STATe?
Query Parameters	None
Query Response	<boolean> = 0 or 1
Description	The Calibration Secure State command enables or disables calibration security. The module powers up with the secure state on. While on, no stores to non-volatile are allowed. In order to turn the state to off the current security code must be supplied. To turn on, the security code does not need to be supplied. If it is supplied the code is checked and a parameter error will occur if it is incorrect. The security code must be supplied in block format. A block format has four parts. The parts are: (1) # (2) a single digit that tells how many digits are in the length (3) the length of the block (4) the actual data, in this case the character of the password (case sensitive) 0 or OFF means values may be stored in non-volatile memory. 1 or ON means values may not be stored in non-volatile memory; this is the default value.  **Calibration commands should only be executed by qualified personnel. Changing these values incorrectly can cause the instrument to perform improperly**

Examples	**Command / Query**	**Response *(Description)***
	CAL:SEC:STAT OFF,#16VM2616	*(Disables calibration security so changes can be made.)*
	CAL:SEC:STAT?	0
	CAL:SEC:STAT ON	*(Enables calibration security.)*
	CAL:SEC:STAT?	ON *(Indicates that the security state is presently enabled.)*

Related Commands	CALibration:SECure:CODE CALibration:STORe

CALibration:STORe

Purpose	Stores the current calibration constants into the non-volatile memory.	
Type	Event	
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 stores the current calibration constants into the non-volatile memory. This command is only effective if the CALibration:SECure:STATe has been disabled.  **Calibration commands should only be executed by qualified personnel. Changing these values incorrectly can cause the instrument to perform improperly**	
Examples	**Command / Query**	**Response *(Description)***
	CAL:SEC:STAT OFF,#16VM2616	*(Disables calibration security so changes can be made.)*
	CAL:STOR	*(This stores the current settings in the non-volatile memory.)*
	CAL:SEC:STAT ON	*(Enables calibration security.)*
Related Commands	CALibration:COUNt? CALibration <channel>: GAIN <value> CALibration <channel>: ZERO <value>	

CALibration:ZERO

Purpose	Sets the selected channel to zero value.
Type	Setting
Command Syntax	CALibration <channel>:ZERO <value>
Command Parameters	<channel> = 1 to 16 (VM2616) or 1 to 8 (VM2608) <value> = -128 to +127
***RST Value**	N/A
Query Syntax	CALibration <channel>:ZERO?
Query Parameters	<channel > = 1 to 16 (VM2616) or 1 to 8 (VM2608)
Query Response	<value> = -128 to +127
Description	The Calibration Zero command sets 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 commands should only be executed by qualified personnel. Changing these values incorrectly can cause the instrument to perform improperly**

Examples	**Command / Query**	**Response *(Description)***
	CAL:SEC:STAT OFF,#16VM2616	*(Disables calibration security so changes can be made.)*
	CAL1:ZERO 50	
	CAL:SEC:STAT ON	*Enables calibration security.*
	CAL1:ZERO?	50

Related Commands	CALibration:DEFault CALibration <channel>: GAIN <value> CALibration:STORe

FETCh:AVErage?

Purpose	Retrieves the average value out of a group of data collected on the previous run.
Type	Query
Command Syntax	None - Query only
Command Parameters	N/A
***RST Value**	N/A
Query Syntax	FETCh:AVErage? [<fetch_counts> [, <fetch_address>][, <channel_list>]]
Query Parameters	<fetch_counts> = Numeric ASCII value <fetch_address> = Numeric ASCII value <channel_list> = Numeric ASCII value
Query Response	Numeric ASCII value
Description	The Fetch Average Query command specifies what data is to be searched by giving a starting point the <fetch_address> and the number of points <fetch_counts>. The value is returned for each channel specified in the <channel_list>. When multiple channels are specified, the values are separated by commas. The return value is the average in the specified data set. <fetch_counts> How many data points are to be retrieved. A fetch count of 0 is from the fetch address to the end of memory. <fetch_address> Specifies the starting point for the data retrieval. A value of 0 represents the trigger point, a - value represents pre-trigger information and a + value represents post-trigger information. <channel_list> Standard channel list format supporting Channels 1 through 16 (VM2616) or 1 through 8 (VM2608). If the channel list is not supplied, all channels are used.

Examples	**Command / Query**	**Response *(Description)***
	FETC:AVE? (@1)	1.111111 *(This queries the average on channel one over all the data collected.)*

Related Commands	FETCh:DATA? FETCh:MAXimum? FETCh:MINimum? FETCh:NTRansition? FETCh:PP? FETCh:PTRansition? FETCh:TRMS

FETCh:DATA?

Purpose	Retrieves the data collected from the previous run.
Type	Query
Command Syntax	None - Query only
Command Parameters	N/A
***RST Value**	N/A
Query Syntax	FETCh:DATA? [<fetch_counts> [, <fetch_address>][, <channel_list>]]
Query Parameters	<fetch_counts> = Numeric ASCII value <fetch_address> = Numeric ASCII value <channel_list> = Numeric ASCII value
Query Response	Numeric ASCII values
Description	The Fetch Data query specifies what data is to be queried by giving a starting point, <fetch_address>, and the number of points, <fetch_counts>. The value is returned for each channel specified in the <channel_list>. When multiple channels are specified, the values are separated by commas. <fetch_counts> How many data points are to be retrieved. A fetch count of 0 is from the fetch address to the end of memory. <fetch_address> Specifies the starting point for the data retrieval. A value of 0 represents the trigger point, a - value represents pre-trigger information and a + value represents post-trigger information. <channel_list> Standard channel list format supporting Channels 1 through 16 (VM2616) or 1 through 8 (VM2608). If the channel list is not supplied, all channels are used.

Examples	**Command / Query**	**Response *(Description)***
	FETC:DATA? 3,-2,(@5)	0.000000,1.111111,2.222222 *(This example fetches three data values on channel five, starting at two before the trigger point.)*

Related Commands	FETCh:AVErage? FETCh:MAXimum? FETCh:MINimum? FETCh:NTRansition? FETCh:PP? FETCh:PTRansition? FETCh:TRMS?

FETCh:MAXimum?

Purpose	Retrieves the maximum value out of a group of data collected during the previous run.
Type	Query
Command Syntax	None - Query only
Command Parameters	N/A
***RST Value**	N/A
Query Syntax	FETCh:MAXimum? [<fetch_counts>[,<fetch_address>][,<channel_list>]]
Query Parameters	<fetch_counts> = Numeric ASCII value <fetch_address> = Numeric ASCII value <channel_list> = Numeric ASCII value
Query Response	Numeric ASCII value
Description	The Fetch Maximum query specifies that the maximum value out of a group of data collected over the previous run is to be returned. The data set in which the query is to be performed, is specified by giving a starting point, <fetch_address>, and the number of points, <fetch_counts>. The values are returned for all the channels specified in the channel list. When multiple channels are specified, the values are separated by commas. The return value is the maximum in the specified data. <fetch_counts> How many data points are to be retrieved. A fetch count of 0 is from the fetch address to the end of memory. <fetch_address> Specifies the starting point for the data retrieval. A value of 0 represents the trigger point, a - value represents pre-trigger information and a + value represents post-trigger information. <channel_list> Standard channel list format supporting Channels 1 through 16 (VM2616) or 1 through 8 (VM2608). If the channel list is not supplied, all channels are used.

Examples	**Command / Query**	**Response *(Description)***
	FETC:MAX? (@1)	1.111111 *(Returns the maximum on channel one over all the data collected.)*

Related Commands	FETCh:AVErage? FETCh:DATA? FETCh:MINimum? FETCh:NTRansition? FETCh:PP? FETCh:PTRansition? FETCh:TRMS?

FETCh:MINimum?

Purpose	Retrieves the minimum value out of a group of data collected during the previous run.
Type	Query
Command Syntax	None - Query only
Command Parameters	N/A
***RST Value**	N/A
Query Syntax	FETCh:MINimum? [<fetch_counts>[,<fetch_address>][,<channel_list>]]
Query Parameters	<fetch_counts> = Numeric ASCII value <fetch_address> = Numeric ASCII value <channel_list> = Numeric ASCII value
Query Response	Numeric ASCII value
Description	The Fetch Minimum query specifies that the maximum value out of a group of data collected over the previous run is to be returned. The data set in which the query is to be performed, is specified by giving a starting point, <fetch_address>, and the number of points, <fetch_counts>. The values are returned for all the channels specified in the channel list. When multiple channels are specified, the values are separated by commas. The return value is the minimum in the specified data. <fetch_counts> How many data points are to be retrieved. A fetch count of 0 is from the fetch address to the end of memory. <fetch_address> Specifies the starting point for the data retrieval. A value of 0 represents the trigger point, a - value represents pre-trigger information and a + value represents post-trigger information. <channel_list> Standard channel list format supporting Channels 1 through 16 (VM2616) or 1 through 8 (VM2608). If the channel list is not supplied, all channels are used.

Examples	**Command / Query**	**Response *(Description)***
	FETC:MIN? (@1)	1.111111 *(Returns the minimum on channel one over all the data collected.)*

Related Commands	FETCh:AVErage? FETCh:DATA? FETCh:MAXimum? FETCh:NTRansition? FETCh:PP? FETCh:PTRansition? FETCh:TRMS?

FETCh:NTRansition?

Purpose	Retrieves the largest negative transition out of a group of data collected during the previous run.
Type	Query
Command Syntax	None - Query only
Command Parameters	N/A
***RST Value**	N/A
Query Syntax	FETCh:NTRansition? [<fetch_counts> [, <fetch_address>][, <channel_list>]]
Query Parameters	<fetch_counts> = Numeric ASCII value <fetch_address> = Numeric ASCII value <channel_list> = Numeric ASCII value
Query Response	Numeric ASCII value
Description	The Fetch N Transition query returns the largest negative transition value out of a group of data collected over the previous run. The data set in which the query is to be performed, is specified by giving a starting point, <fetch_address>, and the number of points, <fetch_counts>. The values are returned for all the channels specified in the channel list. When multiple channels are specified, the values are separated by commas. The return value is the largest negative transition in the specified data. <fetch_counts> How many data points are to be retrieved. A fetch count of 0 is from the fetch address to the end of memory. <fetch_address> Specifies the starting point for the data retrieval. A value of 0 represents the trigger point, a - value represents pre-trigger information and a + value represents post-trigger information. <channel_list> Standard channel list format supporting Channels 1 through 16 (VM2616) or 1 through 8 (VM2608). If the channel list is not supplied, all channels are used.

Examples	**Command / Query**	**Response *(Description)***
	FETC:NTR? (@1)	1.111111,7 *(This queries and returns the largest negative transition on channel one over all the data collected. 1.111111 is the actual transition value; 7 is the address location of the transition.)*

Related Commands	FETCh:AVErage? FETCh:DATA? FETCh:MAXimum? FETCh:MINimum? FETCh:PP? FETCh:PTRansition? FETCh:TRMS?

FETCh:PP?

Purpose	Retrieves the peak to peak value out of a group of data collected during the previous run.
Type	Query
Command Syntax	None - Query only
Command Parameters	N/A
***RST Value**	N/A
Query Syntax	FETCh:PP? [<fetch_counts> [, <fetch_address>][, <channel_list>]]
Query Parameters	<fetch_counts> = Numeric ASCII value <fetch_address> = Numeric ASCII value <channel_list> = Numeric ASCII value
Query Response	Numeric ASCII value
Description	The Fetch PP query specifies that the peak to peak value out of a group of data collected over the previous run is to be returned. The data set in which the query is to be performed, is specified by giving a starting point, <fetch_address>, and the number of points, <fetch_counts>. The values are returned for all the channels specified in the channel list. When multiple channels are specified, the values are separated by commas. The return value is the peak to peak in the specified data. <fetch_counts> How many data points are to be retrieved. A fetch count of 0 is from the fetch address to the end of memory. <fetch_address> Specifies the starting point for the data retrieval. A value of 0 represents the trigger point, a - value represents pre-trigger information and a + value represents post-trigger information. <channel_list> Standard channel list format supporting Channels 1 through 16 (VM2616) or 1 through 8 (VM2608). If the channel list is not supplied, all channels are used.

Examples	**Command / Query**	**Response *(Description)***
	FETC:PP? (@1)	1.111111 *(This queries the peak to peak value on channel one over all the data collected.)*

Related Commands	FETCh:AVErage? FETCh:DATA? FETCh:MAXimum? FETCh:MINimum? FETCh:NTRansition? FETCh:PTRansition? FETCh:TRMS?

FETCh:PTRansition?

Purpose	Retrieves the largest positive transition out of a group of data collected during the previous run.
Type	Query
Command Syntax	None - Query only
Command Parameters	N/A
***RST Value**	N/A
Query Syntax	FETCh:PTRansition? [<fetch_counts> [,<fetch_address>][,<channel_list>]]
Query Parameters	<fetch_counts> = Numeric ASCII value <fetch_address> = Numeric ASCII value <channel_list> = Numeric ASCII value
Query Response	Numeric ASCII value
Description	The Fetch P Transition query returns the largest positive transition value out of a group of data collected over the previous run. The data set in which the query is to be performed, is specified by giving a starting point, <fetch_address>, and the number of points, <fetch_counts>. The values are returned for all the channels specified in the channel list. When multiple channels are specified, the values are separated by commas. The return value is the largest positive transition in the specified data. <fetch_counts> How many data points are to be retrieved. A fetch count of 0 is from the fetch address to the end of memory. <fetch_address> Specifies the starting point for the data retrieval. A value of 0 represents the trigger point, a - value represents pre-trigger information and a + value represents post-trigger information. <channel_list> Standard channel list format supporting Channels 1 through 16 (VM2616) or 1 through 8 (VM2608). If the channel list is not supplied, all channels are used.

Examples	**Command / Query**	**Response *(Description)***
	FETC:PTR? (@1)	1.111111 *(This queries and returns the largest positive transition on channel one over all the data collected. 1.111111 is the actual transition value; 7 is the address location of the transition.)*

Related Commands	FETCh:AVErage? FETCh:DATA? FETCh:MAXimum? FETCh:MINimum? FETCh:NTRansition? FETCh:PP? FETCh:TRMS?

FETCh:TRMS?

Purpose	Retrieves the true RMS value out of a group of data collected during the previous run.
Type	Query
Command Syntax	None - Query only
Command Parameters	N/A
***RST Value**	N/A
Query Syntax	FETCh:TRMS? [<fetch_counts> [, <fetch_address>][, <channel_list>]]
Query Parameters	<fetch_counts> = Numeric ASCII value <fetch_address> = Numeric ASCII value <channel_list> = Numeric ASCII value
Query Response	Numeric ASCII value
Description	The Fetch TRMS Query specifies that the true RMS value out of a group of data collected over the previous run is to be returned. The data set in which the query is to be performed, is specified by giving a starting point, <fetch_address>, and the number of points, <fetch_counts>. The values are returned for all the channels specified in the channel list. When multiple channels are specified, the values are separated by commas. The return value is the true RMS in the specified data. <fetch_counts> How many data points are to be retrieved. A fetch count of 0 is from the fetch address to the end of memory. <fetch_address> Specifies the starting point for the data retrieval. A value of 0 represents the trigger point, a - value represents pre-trigger information and a + value represents post-trigger information. <channel_list> Standard channel list format supporting Channels 1 through 16 (VM2616) or 1 through 8 (VM2608). If the channel list is not supplied, all channels are used.

Examples	**Command / Query**	**Response *(Description)***
	FETC:TRMS? (@1)	1.111111 *(Returns the true RMS value on channel one over all the data collected.)*

Related Commands	FETCh:AVErage? FETCh:DATA? FETCh:MAXimum? FETCh:MINimum? FETCh:NTRansition? FETCh:PP? FETCh:PTRansition?

FORMat

Purpose	Sets the calibration constant output format.	
Type	Setting	
Command Syntax	FORMat <format_type>	
Command Parameters	<format_type> = ASCII | HEXadecimal | OCTal | BINary	
***RST Value**	ASCII	
Query Syntax	FORMat?	
Query Parameters	None	
Query Response	ASC | HEX | OCT | BIN	
Description	The Format command sets the format of the returned data from the instrument for calibration constant queries.	
Examples	**Command / Query**	**Response *(Description)***
	FORM ASC	*(Sets the output to be in ASCII.)*
	FORM?	ASC
	FORM HEX	*(Sets the output to be in HEXadecimal.)*
	FORM?	HEX
	FORM OCT	*(Sets the output to be in OCTal.)*
	FORM?	OCT
	FORM BIN	*(Sets the output to be in BINary.)*
	FORM?	BIN
Related Commands	CALibration:GAIN? CALibration:ZERO?	

INITiate:DELay

Purpose	Arms the VM2608/2616 after the selected number of sample points have been taken after the command is received.	
Type	Event	
Command Syntax	INITiate:DELay <sample_points>	
Command Parameters	<sample_points> = 1 up to max memory (128k or 512k)	
***RST Value**	1	
Query Syntax	INITiate:DELay?	
Query Parameters	None	
Query Response	Numeric ASCII value	
Description	The Initiate Delay command arms the VM2608/2616 after the selected number of sample points have been taken, after the command is received. This command is used to guarantee that the pre-trigger information is valid. The Initiate Delay command must be followed by an INITiate:[IMMediate] command to arm the device. The Initiate Delay command only sets the delay before the arming.	
Examples	**Command / Query**	**Response *(Description)***
	INIT:DEL 25;IMM	*(First sets the delay then issues an initiate command.)*
	INIT:DEL?	25
Related Commands	ABORt INITiate[:IMMediate]	

INITiate[:IMMediate]

Purpose	Arms the VM2608/2616 upon receipt of the command.	
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 VM2608/2616 upon receipt of the command.	
Examples	**Command / Query**	**Response *(Description)***
	INIT:IMM	
Related Commands	ABORt INITiate:DELay	

OUTPut:TRIGger:SLOPe

Purpose	Sets the active slope of the trigger driven onto the TTL trigger bus.	
Type	Setting	
Command Syntax	OUTPut:TRIGger:SLOPe <slope>	
Command Parameters	<slope> = POSitive | NEGative	
***RST Value**	NEG	
Query Syntax	OUTPut:TRIGger:SLOPe?	
Query Parameters	None	
Query Response	POS | NEG	
Description	The Output Trigger Slope command sets the active slope of the trigger driven onto the TTL trigger bus. The negative slope is the default condition.	
Examples	**Command / Query**	**Response *(Description)***
	OUTP:TRIG:SLOP POS	*(Selects the positive slope of the trigger for the TTL trigger bus.)*
	OUTP:TRIG:SLOP?	POS
	OUTP:TRIG:SLOP NEG	*(Selects the negative slope of the trigger for the TTL trigger bus.)*
	OUTP:TRIG:SLOP?	NEG
Related Commands	OUTPut:TRIGger:TTLTrig OUTPut:TTLTrig:STATe	

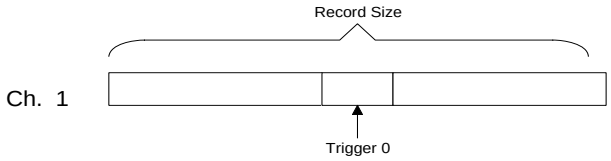
OUTPut:TRIGger:TTLTrig

Purpose	Selects which VXIbus TTL trigger line the module will drive when the output is enabled.	
Type	Setting	
Command Syntax	OUTPut:TRIGger:TTLTrig <n>	
Command Parameters	<n> = 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7	
***RST Value**	TTLT0	
Query Syntax	OUTPut:TRIGger:TTLTrig?	
Query Parameters	None	
Query Response	0 | 1 | 2 | 3 | 4 | 5 | 6 | 7	
Description	The Output Trigger TTL Trig command selects which VXIbus TTL trigger line the module will drive when the output is enabled.	
Examples	**Command / Query**	**Response *(Description)***
	OUTP:TRIG:TTLT 1 OUTP:TRIG:TTLT?	*(Selects TTL trigger 1 as output.)* 1
Related Commands	OUTPut:TRIGger:SLOPe OUTPut:TTLTrig:STATe	

OUTPut:TTLTrig[:STATe]

Purpose	Enables or disables driving the trigger signal onto the VXIbus backplane TTL trigger lines.	
Type	Setting	
Command Syntax	OUTPut:TTLTrig[:STATe] <state>	
Command Parameters	<state> = ON | OFF | 1 | 0	
***RST Value**	0	
Query Syntax	OUTPut:TTLTrig[:STATe]?	
Query Parameters	None	
Query Response	0 | 1	
Description	The Output TTL Trigger State command enables or disables the driving of the trigger signal onto the VXIbus backplane TTL trigger lines. ON would enable the driving while an OFF would disable it.	
Examples	**Command / Query**	**Response *(Description)***
	OUTP:TTLT ON	*(Enables the TTL trigger bus outputs.)*
	OUTP:TTLT?	ON
	OUTP:TTLT OFF	*(Disables the TTL trigger bus outputs.)*
	OUTP:TTLT?	OFF
Related Commands	OUTPut:TRIGger:SLOPe OUTPut:TRIGger:TTLTrig	

REGister:ADDRess

Purpose	Determines where in the record length data will start to be accessed by the registers.
Type	Setting
Command Syntax	REGister:ADDRess <address> [, <channel_list>]
Command Parameters	<address> = Numeric ASCII value <channel_list> = Numeric ASCII value
***RST Value**	Channel 1
Query Syntax	REGister:ADDRess? <channel_list>
Query Parameters	<channel_list> = Numeric ASCII value
Query Response	Numeric ASCII values
Description	The VM2608/2616 supports pseudo-register access, and the Register Address command sets up where in the record length data will start to be accessed by the registers. The <address> value is 0 for the trigger point, and is incremented or decremented by 1, to the allowable value determined by the record size.  Once the register address point is determined, data can be accessed in register format over the VXIbus. The *OPC? command should be used before data is accessed to ensure all word serial commands have been processed. <address> 0 = trigger point -1 = trigger point -1 (up to record size) 1 = trigger point +1 (up to record size) <channel_list> (VM2616) Standard channel list format supporting Channels 1 through 16 or 1 through 8 (VM2608). If the channel list is not supplied, all channels are used.

Examples	**Command / Query**	**Response *(Description)***
	REG:ADDR 0,(@1,2,3)	*(Sets the register address to the trigger point for Channels 1 through 3.)*
	REG:ADDR? (@1,2,3)	0,0,0
Related Commands	None	

ROUTe:CLOSe

Purpose	Enables the channels in the channel list.
Type	Setting
Command Syntax	ROUTe:CLOSe <channel_list>
Command Parameters	<channel_list> = Numeric ASCII value
***RST Value**	All closed
Query Syntax	ROUTe:CLOSe? <channel_list>
Query Parameters	<channel_list> = Numeric ASCII value
Query Response	Numeric ASCII value
Description	The Route Close command enables the channels in the channel list. Enabled channels will record data and will require memory resources. Each group of 4 channels has its own memory associated with the group. The available memory per group is evenly allocated between the channels that are routed closed. The following channels are also or'ed together: 1, 5, 9, 13 2, 6, 10, 14 3, 7, 11, 15 4, 8, 12, 16 Therefore, if Channel 1 and Channel 6 are routed closed, the memory allocated for the first group (Channels 1, 2, 3, and 4), and the second group (Channels 5, 6, 7, and 8), would be allocated to two channels. <channel_list> Standard channel list format supporting Channels 1 through 16 (VM2616) or 1 through 8 (VM2608). If the channel list is not supplied, all channels are used.

Examples	**Command / Query**	**Response *(Description)***
	ROUT:CLOS (@1,2,4)	*(Enables Channels 1, 2 and 4 for data recording.)*
	ROUT:CLOS?	1,2,4
Related Commands	ROUTe:OPEN <channel_list>	

ROUTe:OPEN

Purpose	Disables the channels in the channel list.	
Type	Setting	
Command Syntax	ROUTe:OPEN <channel_list>	
Command Parameters	<channel_list> = Numeric ASCII value	
***RST Value**	All closed	
Query Syntax	ROUTe:OPEN? <channel_list>	
Query Parameters	<channel_list> = Numeric ASCII value	
Query Response	Numeric ASCII value	
Description	The Route Open command disables the channels in the channel list. Disabled channels will not record data and will not use memory resources. <channel_list> Standard channel list format supporting Channels 1 through 16 (VM2616) or 1 through 8 (VM2608). If the channel list is not supplied, all channels are used.	
Examples	**Command / Query**	**Response *(Description)***
	ROUT:OPEN (@1,2,3)	*(Disables Channels 1, 2, and 3 from collecting data and using memory resources.)*
	ROUT:OPEN?	1,2,3
Related Commands	ROUTe:CLOSe <channel_list>	

SWEep:POINts

<table>
<tr><td>Purpose</td><td>Sets the number of points in a record.</td></tr>
<tr><td>Type</td><td>Setting</td></tr>
<tr><td>Command Syntax</td><td>SWEep:POINts <points></td></tr>
<tr><td>Command Parameters</td><td><points> = Numeric ASCII value</td></tr>
<tr><td>*RST Value</td><td>32768 | 131072 (1/4 of total memory size of 128k of 512k)</td></tr>
<tr><td>Query Syntax</td><td>SWEep:POINts?</td></tr>
<tr><td>Query Parameters</td><td>None</td></tr>
<tr><td>Query Response</td><td>Numeric ASCII value</td></tr>
<tr><td>Description</td><td>The Sweep Points command sets the number of points in a record. The number of points must be N, where N ranges from two (2) to the maximum memory allowable:
<table>
<tr><th>Active Channels</th><th>Standard Memory</th><th>512k Word Options</th></tr>
<tr><td>1</td><td>128k Word (131,071)</td><td>512k Words (524,287)</td></tr>
<tr><td>2</td><td>64k Words (64,535)</td><td>256k Words (262,143)</td></tr>
<tr><td>3</td><td>42.6k Words (43,689)</td><td>170.6k Words (174,761)</td></tr>
<tr><td>4</td><td>32k Words (32,767)</td><td>128k Words (131,071)</td></tr>
</table>
All channels acquire data simultaneously, so the number of sweep points set would apply to all channels.</td></tr>
<tr><td>Examples</td><td>
<table>
<tr><th>Command / Query</th><th>Response (Description)</th></tr>
<tr><td>SWE:POIN 25000</td><td>(Sets the sweep points to 25,000 samples.)</td></tr>
<tr><td>SWE:POIN?</td><td>25000</td></tr>
</table>
</td></tr>
<tr><td>Related Commands</td><td>SWEep:STEP</td></tr>
</table>

SWEep:STEP

Purpose	Sets the time interval between samples.	
Type	Setting	
Command Syntax	SWEep:STEP <time>	
Command Parameters	<time> = 0.000010 to 0.1048575 seconds	
***RST Value**	0.000010 seconds	
Query Syntax	SWEep:STEP?	
Query Parameters	None	
Query Response	Floating-point number	
Description	The Sweep Step command sets the time interval between samples. Time is specified in seconds as a floating-point number. This is the sampling rate of the digitizer and is the same for all channels. The sampling rate can be set from 100 kHz (0.000010) to 9.5367 Hz (0.1048575).	
Examples	**Command / Query**	**Response *(Description)***
	SWE:STEP 0.000010 SWE:STEP?	*(Sets the sampling rate to 100 kHz.)* 0.000010
Related Commands	SWEep:POINts	

SYNC

Purpose	Allows synchronization between multiple VM2608/2616 modules.	
Type	Setting	
Command Syntax	SYNC <mode>	
Command Parameters	<mode> = OUTPut0 | OUTPut1 | OUTPut2 | OUTPut3 | OUTPut4 | OUTPut5 | OUTPut6 | OUTPut7 | INPut0 | INPut1 | INPut2 | INPut3 | INPut4 | INPut5 | INPut6 | INPut7 | OFF	
***RST Value**	OFF	
Query Syntax	SYNC?	
Query Parameters	None	
Query Response	OUTPut0 | OUTPut1 | OUTPut2 | OUTPut3 | OUTPut4 | OUTPut5 | OUTPut6 | OUTPut7 | INPut0 | INPut1 | INPut2 | INPut3 | INPut4 | INPut5 | INPut6 | INPut7 | OFF	
Description	The Sync command allows synchronization of multiple VM2608/2616 modules in a VXIbus chassis. For multiple VM2608/2616 modules to simultaneously acquire data, each card is programmed to trigger off a VXIbus trigger line, and one card generates a sync pulse to that trigger line. Up to 576 channels, in a 13-slot chassis, can be programmed to simultaneously acquire data.	
Examples	**Command / Query**	**Response *(Description)***
	SYNC:OUTP1	*(This sets the trigger pulse to be generated from OUTPut1)*
	SYNC?	OUTP1
	SYNC OFF	*(This turns off trigger generation from the card. This is the *RST value.)*
	SYNC?	OFF
Related Commands	None	

TRACe:LENGth?

Purpose	Returns actual size of the data captured during the previous run.	
Type	Query	
Command Syntax	None - Query only	
Command Parameters	N/A	
***RST Value**	N/A	
Query Syntax	TRACe:LENGth?	
Query Parameters	None	
Query Response	Numeric ASCII value	
Description	Returns the actual size of data captured during the previous run.	
Examples	**Command / Query**	**Response *(Description)***
	TRAC:LENG?	50
Related Commands	TRACe:POINts?	

TRACe:POINts?

Purpose	Queries the potential length that can be captured based on current settings.	
Type	Query	
Command Syntax	None - Query only	
Command Parameters	N/A	
***RST Value**	N/A	
Query Syntax	TRACe:POINts?	
Query Parameters	None	
Query Response	Numeric ASCII value	
Description	Returns the length of the potential trace that can be captured based upon the current settings.	
Examples	**Command / Query**	**Response *(Description)***
	TRAC:POIN?	32768
Related Commands	TRACe:LENGth	

TRIGger[:IMMediate]

Purpose	Word serial trigger command to initiate data capture.	
Type	Event	
Command Syntax	TRIGger[:IMMediate]	
Command Parameters	None	
***RST Value**	N/A	
Query Syntax	None	
Query Parameters	N/A	
Query Response	N/A	
Description	The Trigger Immediate command performs the same function as the ****TRG*** command. This command takes effect only when the module is armed for a trigger.	
Examples	**Command / Query**	**Response *(Description)***
	TRIG	
Related Commands	INITiate:DELay INITiate[:IMMediate] *TRG	

TRIGger:LEVel

Purpose	Sets the trigger level for the selected trigger source.	
Type	Setting	
Command Syntax	TRIGger:LEVel <trigger_level>	
Command Parameters	<trigger_level> = Numeric ASCII value	
***RST Value**	0.0 V	
Query Syntax	TRIGger:LEVel?	
Query Parameters	None	
Query Response	Numeric ASCII value	
Description	The Trigger Level command sets the trigger level for the selected trigger source, given that the source is either the front panel external inputs, or one of the measurement inputs. The valid range for the trigger level is the entire input range at the selected gain (i.e. the 1-volt range can have a trigger level from -1.0 volts to +1.0 volts). The external input has a range of -10 volts to +10 volts.	
Examples	**Command / Query**	**Response *(Description)***
	TRIG:LEV 0.5 TRIG:LEV?	 0.5
Related Commands	TRIGger:SLOPe TRIGger:SOURce VOLTage:RANGe	

TRIGger:SLOPe

Purpose	Selects the active edge for triggering the VM2608/2616.	
Type	Setting	
Command Syntax	TRIGger:SLOPe <slope>	
Command Parameters	<slope> = POSitive | NEGative	
***RST Value**	NEG	
Query Syntax	TRIGger:SLOPe?	
Query Parameters	None	
Query Response	POS | NEG	
Description	The Trigger Slope command selects the active edge for triggering the VM2608/2616. Selecting the positive slope will require that the trigger input make a negative to positive transition through the trigger level, while a negative slope requires the trigger input to make a positive to negative transition.	
Examples	**Command / Query**	**Response *(Description)***
	TRIG:SLOP POS	
	TRIG:SLOP?	POS
	TRIG:SLOP NEG	
	TRIG:SLOP?	NEG
Related Commands	TRIGger:LEVel TRIGger:SOURce	

TRIGger:SOURce

Purpose	Selects the source for triggering the VM2608/2616.
Type	Setting
Command Syntax	TRIGger:SOURce <source>
Command Parameters	<source> = CH1 | CH2 | CH3 | CH4 | CH5 | CH6 | CH7 | CH8 | CH[9-16] | EXTernal | TTLT[0-7] | IMMediate
***RST Value**	CH1
Query Syntax	TRIGger:SOURce?
Query Parameters	None
Query Response	CH1 | CH2 | CH3 | CH4 | CH5 | CH6 | CH7 | CH8 | CH[9-16] | EXT | TTLT[0-7] | IMM
Description	The Trigger Source command selects the source of input trigger for data capture. The query returns the source of the input trigger used for data capture. Note that Channels 9 through 16 are only available with a VM2616.

Examples

Command / Query	**Response *(Description)***
TRIG:SOUR TTLT0	*(This sets the input trigger source to TTL Trigger 0.)*
TRIG:SOUR?	TLTT0
TRIG:SOUR CH5	*(This sets the input trigger source to Channel 5.)*
TRIG:SOUR?	CH5
TRIG:SOUR EXT	*(This sets the input trigger source to an external source.)*
TRIG:SOUR?	EXT

Related Commands	TRIGger:LEVel TRIGger:SLOPe

VOLTage:RANGe

Purpose	Sets the input range for the channels listed by the channel list.
Type	Setting
Command Syntax	VOLTage:RANGe <range>,<channel list>
Command Parameters	<range> = 0.1 | 0.4 | 1 | 4 | 10 | 40 <channel_list> = Numeric ASCII value
***RST Value**	40.0 V
Query Syntax	VOLTage:RANGe? <channel list>
Query Parameters	<channel_list> = Numeric ASCII value
Query Response	0.1 | 0.4 | 1 | 4 | 10 | 40
Description	The voltage range command sets the input range for the channels described by the channel list. The 40 volt range is both the power up condition and *RST condition. <channel_list> Standard channel list format supporting Channels 1 through 16 (VM2616) or 1 through 8 (VM2608). If the channel list is not supplied, all channels are used.
Examples	**Command / Query** / **Response *(Description)***
	VOLT:RANG 10,(@1:4)
	VOLT:RANG? (@1:4) → 10
Related Commands	None

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.	
Examples	**Command / Query**	**Response *(Description)***
	STAT:OPER:COND?	0
Related Commands	STATus:OPERation:ENABle	

STATus:OPERation:ENABle

Purpose	Sets the Operation Status Enable Register.	
Type	Required SCPI command	
Command Syntax	STATus:OPERation:ENABle <NRf>	
Command Parameters	NRf = 0 to 32767	
***RST Value**	NRf must be specified	
Query Syntax	STATus:OPERation:ENABle?	
Query Parameters	None	
Query Response	0 to 32767	
Description	The Operation Status Enable Register is included for SCPI compatibility and the VM2608/2616 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 VM2608/2616 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 VM2608/2616 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 = 0 to 32767	
***RST Value**	NRf must be supplied	
Query Syntax	STATus:QUEStionable:ENABle?	
Query Parameters	None	
Query Response	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 VM2608/2616 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 VM2608/2616 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 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 version of the SCPI standard with which the module 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 with which the VM2608/2616 complies.	
Examples	**Command / Query**	**Response *(Description)***
	SYST:VERS?	1994.0
Related Commands	None	

SECTION 5

THEORY OF OPERATION

INTRODUCTION

The VM2608/2616 instrumentation module provides 8/16 independent channels of an analog to digital converter (ADC), with 16 bits of resolution and a sample rate of up to 100 kSamples/s. Each channel consists of an independent ADC combined with its own instrumentation amplifier providing a true differential input (see Figure 5-1).

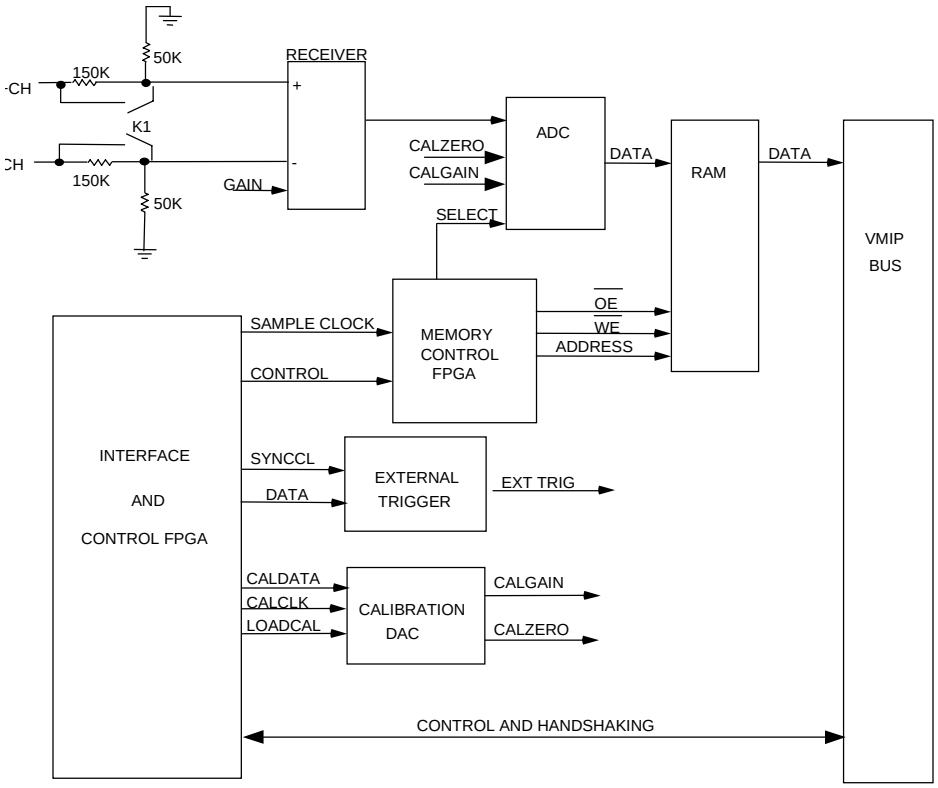


FIGURE 5-1: VM2608/2616 FUNCTIONAL BLOCK DIAGRAM

Each group of four channels has a separate bank of memory that may be allocated to one, two, three or four converters. This allows for the most efficient use of the available memory. The data may also be directly routed to the VXIbus for direct data collection.

The input data is acquired when a conversion trigger event occurs from one of the three following methods:

1) Trigger source from the front panel input: This input has a ±10 volt input range, which can be programmed to trigger on the rising or falling edge of this signal. The voltage level can be specified with a 12-bit resolution and 1% accuracy.
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 ADCs will convert when a word serial command is received by the instrument.
4) Trigger from one of the 8 or 16 input channels: One of the input channels can be selected to trigger the board. It can be programmed to trigger during a rising or falling edge, and at a specified voltage with 12 bits of resolution and 1% of range accuracy.

The trigger event causes all channels to convert simultaneously and pre- and post-triggering is available for added flexibility.

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.

INTERFACE AND CONTROL FPGA

The interface FPGA controls all functions needed to interface with the VMIP host. The interface FPGA receives data from the VMIP bus for the sweep step size and is transferred to the sample clock generation circuitry (see Figure 5-2). This circuitry will generate the specified step size by counting down from the 10 MHz clock of the VXI backplane. The output of this count down chain is the sample clock and is routed to the memory control FPGA that controls the collection of the data from the ADC conversion cycle. The sample size is loaded into a 20-bit sample counter that increments once for each conversion cycle. This counter will disable the sample clock when the counter reaches its top count signifying that all samples have been recorded.

The interface FPGA also contains the trigger logic that monitors the VXI backplane TTL trigger, the external trigger and internal trigger lines. This logic will start the conversion cycle by issuing the SYNCIN signal to the sample clock circuitry when the appropriate trigger has been received. The interface FPGA will also generate a TTL TRIGGER in the specified polarity to the VXI backplane if enabled and in the master mode.

The calibration logic also resides in this FPGA. This logic will receive the calibration logic from the VMIP bus and generate the appropriate clock and control signals to the calibration DACs.

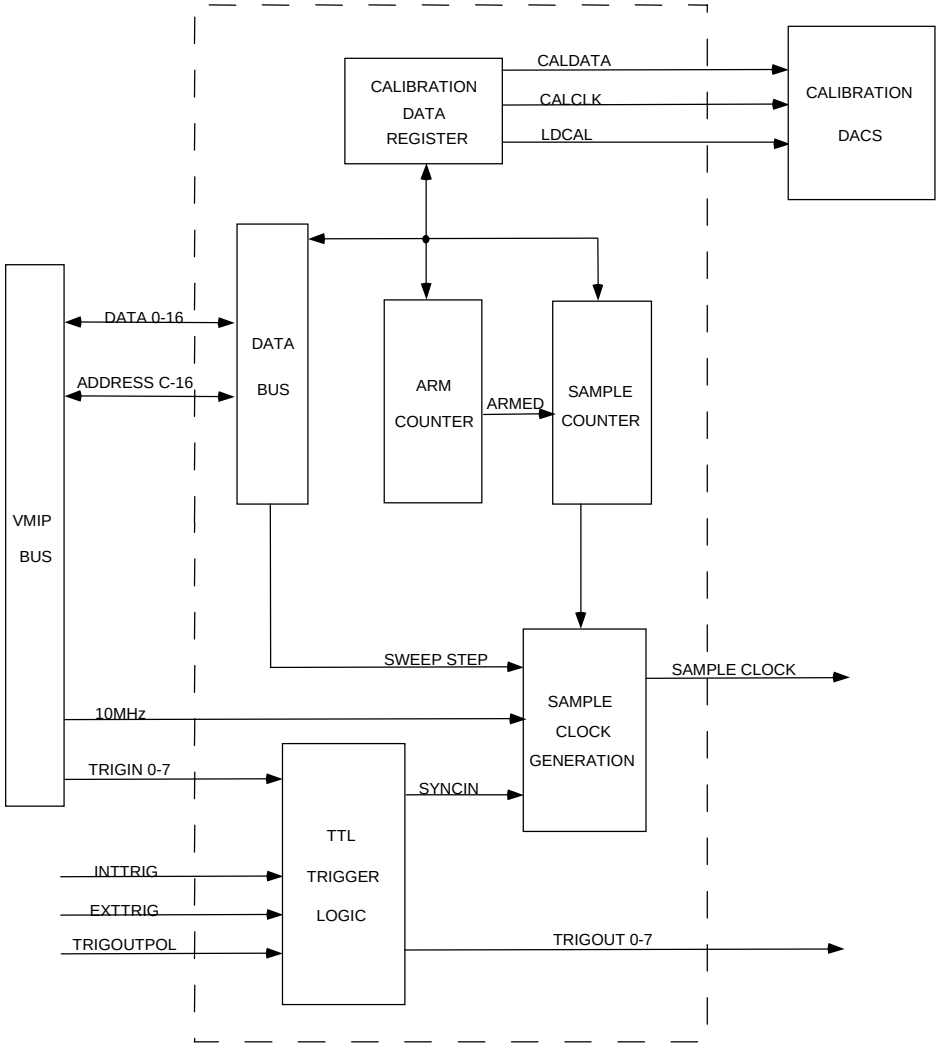


FIGURE 5-2: INTERFACE AND CONTROL FPGA FUNCTIONAL DIAGRAM

DATA CAPTURE

The interface and control FPGA will generate relay data and clocks based on the voltage range specified. If the voltage range is ±0.4, 4.0, or 40.0 V for the input signal, K1 will be open. This will select the voltage divider created by R1 and R2 for the positive input, and R3 and R4 for the negative input (see Figure 5-3). Upon receipt of the specified trigger, the input data is captured from the front panel connectors. The data is initially received by the differential receiver. The output BUFCH is routed to the sixteen bit Analog to Digital Converters.

The memory control FPGA then generates the trigger point address and outputs this address onto the RAM memory address bus. The memory control FPGA then generates the conversion control lines CS* and R/C* based on the sample clock input from the interface and control FPGA. When CS* is low the falling edge of R/C* initiates a new conversion and the rising edge causes the ADC to output parallel data onto the RAM data bus. The memory control FPGA then generates the active low WE* signal writing data into the RAM.

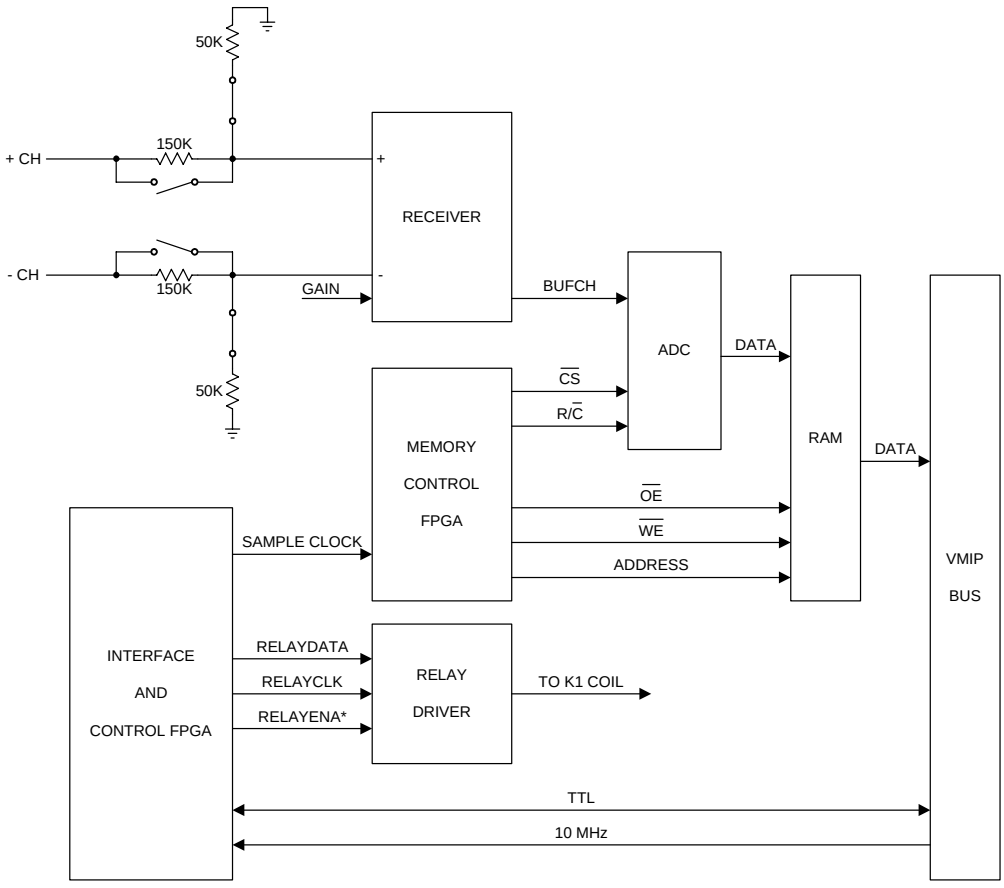


FIGURE 5-3: DATA CAPTURE FUNCTIONAL DIAGRAM

INDEX

T

V

W