

ARTISAN®
TECHNOLOGY GROUP

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

**Artisan Technology Group**
(217) 352-9330 | sales@artisantg.com | artisantg.com

**Full-service, independent repair center**
with experienced engineers and technicians on staff.

**We buy your excess, underutilized, and idle equipment**
along with credit for buybacks and trade-ins.

**Custom engineering**
so your equipment works exactly as you specify.

- Critical and expedited services
- In stock / Ready-to-ship
- Leasing / Rentals / Demos
- ITAR-certified secure asset solutions

**Expert team | Trust guarantee | 100% satisfaction**

All trademarks, brand names, and brands appearing herein are the property of their respective owners.

**Visit our website - *Click HERE***

# VM2716A

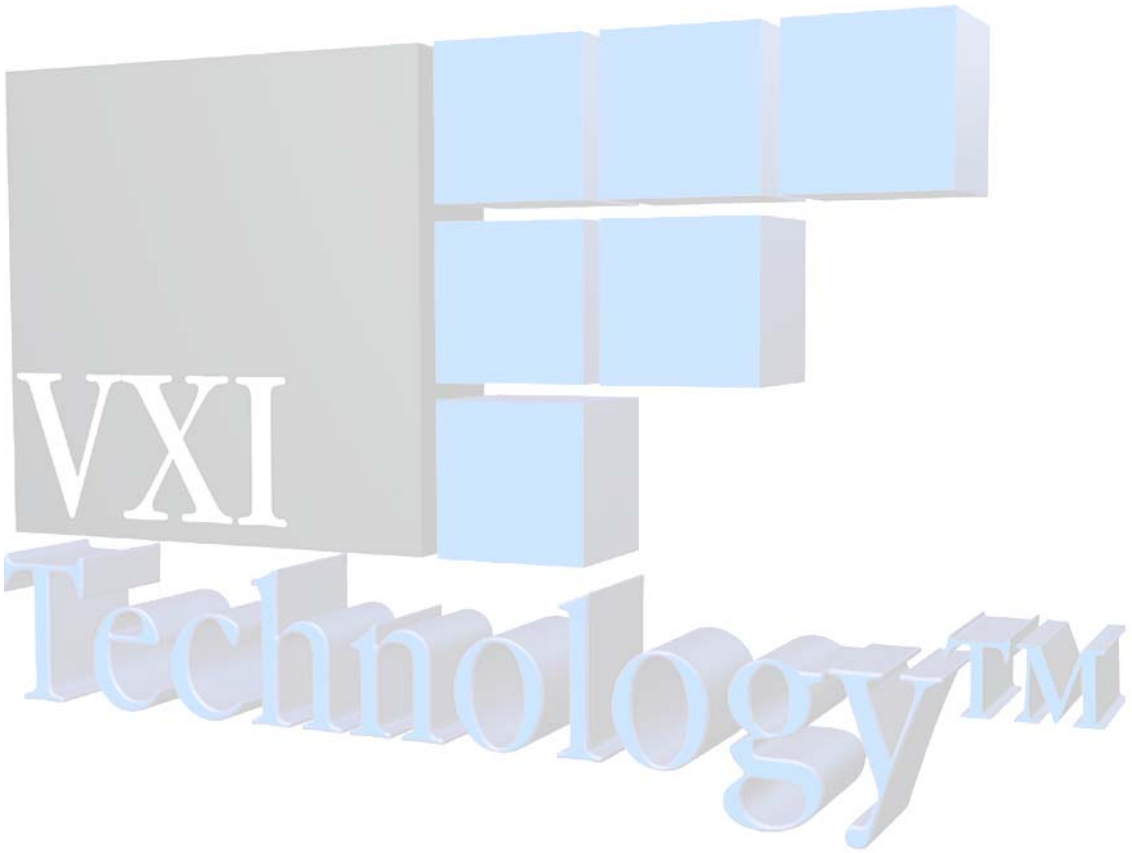
## DIFFERENTIAL SCANNING VOLTMETER

# USER'S MANUAL

**82-0026-000**
**Released April 7, 2003**

**VXI Technology, Inc.**

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


VXI
Technology™

# TABLE OF CONTENTS

**INTRODUCITON**

Certification ..... 6
Warranty ..... 6
Limitation of Warranty ..... 6
Restricted Rights Legend ..... 6
DECLARATION OF CONFORMITY ..... 7
GENERAL SAFETY INSTRUCTIONS ..... 9
Terms and Symbols ..... 9
Warnings ..... 9
SUPPORT RESOURCES ..... 11

**SECTION 1 ..... 13**

INTRODUCTION ..... 13
Introduction ..... 13
Description ..... 14
VM2716A General Specifications ..... 16

**SECTION 2 ..... 17**

PREPARATION FOR USE ..... 17
Installation ..... 17
Calculating System Power and Cooling Requirements ..... 17
Setting the Chassis Backplane Jumpers ..... 18
Setting the Logical Address ..... 18
Front Panel Interface Wiring ..... 19

**SECTION 3 ..... 21**

INTRODUCTION ..... 21
Examples of SCPI Commands ..... 21
ABORT ..... 21
CALIBRATION:COUNT? ..... 22
CALIBRATION:DEFAULT ..... 23
CALIBRATION:GAIN ..... 24
CALIBRATION:RESET ..... 25
CALIBRATION:SECURE:CODE ..... 26
CALIBRATION:SECURE:STATE ..... 27
CALIBRATION:STORE ..... 28
CALIBRATION:ZERO ..... 29
CONFIGURE:[VOLTAGE:]APERTURE ..... 30
CONFIGURE:VOLTAGE ..... 31
FETCH? ..... 32
FORMAT:VOLTAGE:DATA ..... 33
INITIATE ..... 34
MEASURE:VOLTAGE ..... 35
OUTPUT:TTLTRIG ..... 36
OUTPUT:TTLTRIG:POLARITY ..... 37
OUTPUT:TTLTRIG:STATE ..... 38
OUTPUT:TTLTRIG:TYPE ..... 39
SAMPLE:COUNT ..... 40
SAMPLE:TIMER ..... 41
TRACE:DATA:FEED ..... 42
TRIGGER:IMMEDIATE ..... 43
TRIGGER:SEQUENCE:SLOPE ..... 44
TRIGGER:SEQUENCE:SOURCE ..... 45

APPLICATION EXAMPLES ..... 46
REGISTER ACCESS EXAMPLES ..... 48
VXIPLUG&PLAY EXAMPLES ..... 49
**SECTION 4 ..... 55**
COMMAND DICTIONARY ..... 55
Introduction ..... 55
Alphabetical Command Listing ..... 55
Command Dictionary ..... 59
COMMON SCPI COMMANDS ..... 60
*CLS ..... 60
*ESE ..... 61
*ESR? ..... 62
*IDN? ..... 63
*OPC ..... 64
*RST ..... 65
*STB? ..... 66
*TRG ..... 67
*TST? ..... 68
*WAI ..... 69
INSTRUMENT SPECIFIC SCPI COMMANDS ..... 70
ABORt ..... 70
CALibration:COUNt? ..... 71
CALibration:DEFault ..... 72
CALibration:GAIN ..... 73
CALibration:RESet ..... 74
CALibration:SECure:CODE ..... 75
CALibration[:SECure]:STATe ..... 76
CALibration:STORe ..... 77
CALibration:ZERO ..... 78
CONFigure:[VOLTage:]APERture ..... 79
CONFigure:VOLTage ..... 80
DOWNLOAD ..... 81
FETCh? ..... 82
FORMat[:VOLTage][:DATA] ..... 83
INITiate ..... 84
IT ..... 85
J ..... 86
MEASure[:VOLTage] ..... 87
OUTput:TTLTrig ..... 88
OUTput:TTLTrig ..... 88
OUTPut:TTLTrig:POLarity ..... 89
OUTPut[:TTLTrig]:STATe ..... 90
OUTPut:TTLTrig:TYPE ..... 91
SAMPle:COUNt ..... 92
SAMPle:TIMer ..... 93
TRACe[:DATA]:FEED ..... 94
TRIGger[:IMMediate] ..... 95
TRIGger[:SEQuence]:SLOPe ..... 96
TRIGger[:SEQuence]:SOURce ..... 97
TRIGger:TYPE ..... 98
REQUIRED SCPI COMMANDS ..... 99
STATus:OPERation:CONDition? ..... 99
STATus:OPERation:ENABle ..... 100
STATus:OPERation[:EVENt]? ..... 101
STATus:PRESet ..... 102
STATus:QUEStionable:CONDition? ..... 103
STATus:QUEStionable:ENABle? ..... 104

STATus:QUEStionable[:EVENt]?....105
SYSTem:ERRor?....106
SYSTem:VERSion?....107
**SECTION 5....109**
THEORY OF OPERATION....109
Introduction....109
Input and Buffering Circuitry....110
Calibration and measurement circuitry....111
Timing and Control FPGA....113
Power Forms Generation and Regulation....115
**INDEX....117**

## 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<br>Irvine, California 92614-6509 |
| **PRODUCT NAME** | Differential Scanning Voltmeter |
| **MODEL NUMBER(S)** | VM2716A |
| **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<br>CISPR 22 (1997) Class A<br>VCCI (April 2000) Class A<br>ICES-003 Class A (ANSI C63.4 1992)<br>AS/NZS 3548 (w/A1 & A2:97) Class A<br>FCC Part 15 Subpart B Class A<br>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.*

**April 2003**

CE

***Jerry Patton, 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

## WARNINGS

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

**Use Proper Power Cord** To avoid hazard, only use the power cord specified for this product.

**Use Proper Power Source** To avoid electrical overload, electric shock, or fire hazard, do not use a power source that applies other than the specified voltage.

**Use Proper Fuse** To avoid fire hazard, only use the type and rating fuse specified for this product.

## WARNINGS (CONT.)

**Avoid Electric Shock**

To avoid electric shock or fire hazard, do not operate this product with the covers removed Do not connect or disconnect any cable, probes, test leads, etc. while they are connected to a voltage source. Remove all power and unplug unit before performing any service. ***Service should only be performed by qualified personnel.***

**Ground the Product**

This product is grounded through the grounding conductor of the power cord. To avoid electric shock, the grounding conductor must be connected to earth ground.

**Operating Conditions**

To avoid injury, electric shock or fire hazard:

- Do not operate in wet or damp conditions.
- Do not operate in an explosive atmosphere.
- Operate or store only in specified temperature range.
- Provide proper clearance for product ventilation to prevent overheating.
- DO NOT operate if any damage to this product is suspected. ***Product should be inspected or serviced only by qualified personnel.***



**Improper Use**

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

# SUPPORT RESOURCES

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




**Internet Support**

E-mail: support@vxitech.com
Web Address: http://www.vxitech.com

**Telephone Support (U.S.)**

Tel: (949) 955-1894 **West Coast**
(216) 447-8950 **East Coast**

Fax: (949) 955-3041 **West Coast**
(216) 447-8951 **East Coast**

**VXI Technology Headquarters**

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

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

# SECTION 1

# INTRODUCTION

## INTRODUCTION

The VM2716A provides the functionality of a 4.5 digit DC voltmeter and a FET multiplexer. The instrument uses the message-based word serial interface for programming and data movement, as well as supporting direct register access for very high-speed data retrieval. The VM2716A command set conforms to SCPI standards for consistency and ease of programming.

The VM2716A is a member of the VXI Technology VMIP™ (VXI Modular Instrumentation Platform) family and is available as a 16-, 32- or 48-channel, single-wide, C-size VXIbus instrument. In addition to these three standard configurations, the VM2716A may be combined with any of the other members of the VMIP family to form a customized and highly integrated instrument (see Figure 1-1). This allows the user to reduce system size and cost by combining the VM2716A with two other instrument functions in a single-wide C-size VXIbus module. Figure 1-2 shows the 48-channel version of the VM2716A. The 32-channel version would not have J200 and its associated LEDs and nomenclature while the 16-channel version would also eliminate J202.

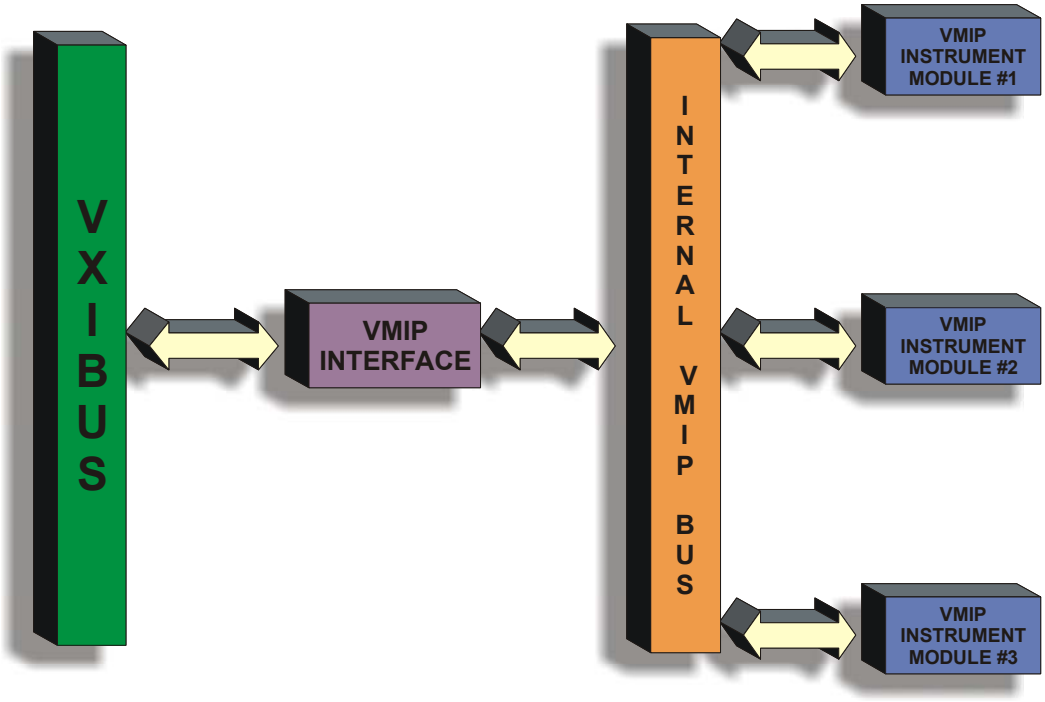
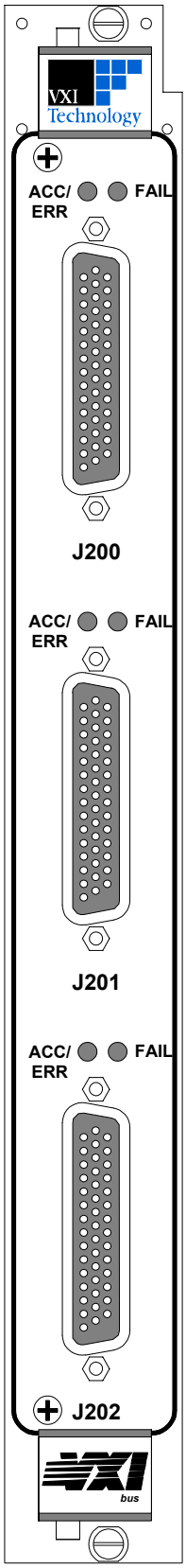



FIGURE 1-1 VMIP™ PLATFORM




Regardless of whether the VM2716A is configured with other VM2716A modules or with other VMIP modules, each group of 16 channels is treated as an independent instrument in the VXIbus chassis, has its own Unique Logical Address and, as such, each group has its own FAIL and ACCESS light.

## DESCRIPTION

The VM2716A provides the functionality of a 4.5 digit DC voltmeter and a FET multiplexer on a single instrumentation module. For applications where multiple voltage points need to be measured, and the overhead and cost of a full digital multimeter (DMM) and additional switch module are not necessary, the VM2716A becomes an ideal instrument.

The VM2716A is part of the VMIP family of instruments and can be combined with up to two other modules to form a high-density VXIbus instrument that fully utilizes the capabilities of the VMIP module. Up to 48 differential channels can be scanned in a single C-size card at a rate of 100 readings per second, at 4.5 digits of resolution.

For applications that require fast DC voltage measurements on many channels, the price/performance value of the VM2716A scanning voltmeter makes it an ideal choice.

The scanning voltmeter is programmed using message-based, word serial protocol. The commands are SCPI and IEEE-STD-488.2 compatible. VXI*plug&play* drivers are also provided to further ease programming. For faster data access the VM2716A also supports direct register data access.

The VM2716A can be programmed to scan through its channel list either upon receipt of a word serial trigger or by programming the internal sample timer from 1 ms to 512 ms, allowing for up to 1000 readings per second at 4.5 digits of resolution.

Aperture times may also be programmed from 400 $\mu$s to 20 ms allowing for the rejection of 50 Hz or 60 Hz noise.

To synchronize to other instruments within the VXIbus chassis the VM2716A can also be programmed to generate a sync pulse to any one of the TTL trigger lines upon completion of operation.

The calibration constants used to correct the data values are stored in non-volatile memory. These constants are determined when an instrument is calibrated and can be changed as necessary (such as during routine calibration cycles). 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.

**FIGURE 1-2 FRONT PANEL LAYOUT**

The block diagram of Figure 1-3 shows the overall functionality of the VM2716A Scanning Voltmeter instrument.

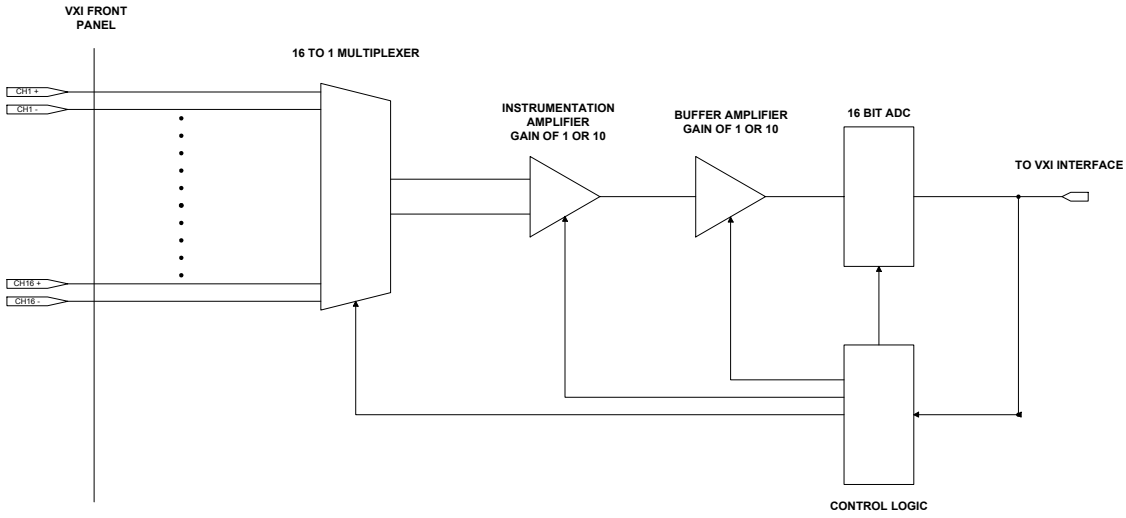



**FIGURE 1-3 VM2716A BLOCK DIAGRAM**

## VM2716A General Specifications

| General Specifications | | | | |
|---|---|---|---|---|
| **Channels** | | | | |
| **VM2716A-1** | 16 | | | |
| **VM2716A-2** | 32 | | | |
| **VM2716A-3** | 48 | | | |
| **Input Range** | | | | |
| | ±0.1 V, ±1.0 V, ±10.0 V Ranges | | | |
| **Maximum Voltage** | | | | |
| | < ±16 V pk (Normal mode plus common mode) | | | |
| **Maximum Common Mode Voltage** | | | | |
| | < ±16 V pk | | | |
| **Resolution** | | | | |
| | 10 $\mu$V in 0.1 V range<br>100 $\mu$V in 1.0 V range<br>1 mV in 10.0 V range | | | |
| **Input Resistance** | | | | |
| | 10 MΩ from any input pin to ground | | | |
| **Input Over-voltage Protection** | | | | |
| | ±250 VDC or peak AC between differential input pairs<br>±125 VDC or peak AC from any input to ground | | | |
| **Accuracy @ $T_{CAL}$ ±5°C** | | | | |
| | ±0.015% of range for 90 days<br>±0.03% of range for 1 year | | | |
| **Reading Rate** | | | | |
| **Aperature** | 20 ms | 16.66 ms | 2.5 ms | 400 $\mu$s |
| **Readings/Second** | 44 | 52 | 298 | 1000 |
| **Power Requirements** | | | | |
| **VM2716A-1** | +5.0 V@1.06 A, +12.0 V@0.01 A, +24.0 V@0.04 A, -24.0 V@0.04 A, -5.2 V@0.12 A | | | |
| **VM2716A-2** | +5.0 V@1.38 A, +12.0 V@0.02 A, +24.0 V@0.08 A, -24.0 V@0.08 A, -5.2 V@0.19 A | | | |
| **VM2716A-3** | +5.0 V@1.70 A, +12.0 V@0.03 A, +24.0 V@0.12 A, -24.0 V@0.12 A, -5.2 V@0.26 A | | | |
| **Manufacturer's ID** | | | | |
| | 3915 | | | |
| **Module Model Code** | | | | |
| | 269 | | | |
| **Cooling Requirements** | | | | |
| | 0.7 L/s | | | |

# SECTION 2

# PREPARATION FOR USE

## INSTALLATION

When the VM2716A is unpacked from its shipping carton, the contents should include the following items:

(1) VM2716A VXIbus module.
(1) VM2716A Differential Scanning Voltmeter User's Manual (this manual).

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

Once the VM2716A is assessed to be in good condition, it may be installed into an appropriate C-size or D-size VXIbus chassis in any slot other than slot 0. The chassis should be checked to ensure that it is capable of providing adequate power and cooling for the VM2716A. Once the chassis is found be adequate, the VM2716A's logical address and the chassis' backplane jumpers should be configured prior to the VM2716A's installation.

## CALCULATING SYSTEM POWER AND COOLING REQUIREMENTS

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



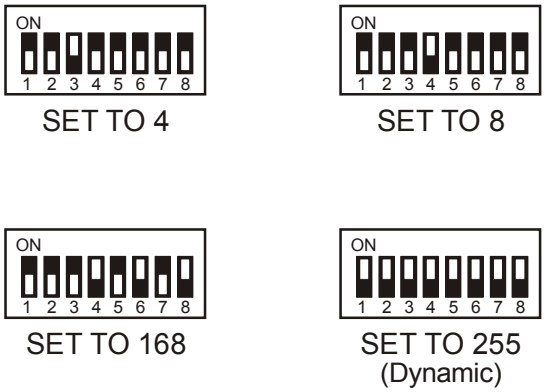
It should be noted that if the chassis cannot provide adequate power to the module, the instrument may not perform to specification or possibly not operate at all. In addition, if adequate cooling is not provided, the reliability of the instrument will be jeopardized and permanent damage may occur. Damage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

## SETTING THE CHASSIS BACKPLANE JUMPERS

Please refer to the chassis User's Manual for further details on setting the backplane jumpers.

## SETTING THE LOGICAL ADDRESS

The logical address of the VM2716A is set by a single 8-position DIP switch located near the module's backplane connectors (this is the only switch on the module). The switch is labeled with positions 1 through 8 and with an ON position. A switch pushed toward the ON legend will signify a logic 1; switches pushed away from the ON legend will signify a logic 0. The switch located at position 1 is the least significant bit while the switch located at position 8 is the most significant bit. See Figure 2-1 for examples of setting the logical address switch.




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

**FIGURE 2-1 LOGICAL ADDRESS SWITCH SETTING EXAMPLES**

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

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

## FRONT PANEL INTERFACE WIRING

The VM2716A's interface is made available on the front panel of the instrument. The 16-channel version (VM2716A-1) will have J201 that contains all signals for this instrument. The 32-channel version (VM2716A-2) will have J201 and J202 provided, while the 48-channel version (VM2716A-3) will have J200, J201 and J202. The wiring for each of these connectors is identical and since each group of 16 channels is treated as a separate instrument, the module will have three Channel 1s, three Channel 2s, three Channel 3s, etc.

The connector used in the VM2716A is a commonly available 44-pin high density DSUB receptacle connector. A mating solder cup pin connector from AMP is included, crimp type connectors are available from a variety of sources.

### TABLE 2-1 SCANNING VOLTMETER PIN OUTS

| SIGNAL | PIN NUMBER | SIGNAL | PIN NUMBER |
|---|---|---|---|
| CHANNEL 1 + | 1 | GROUND | 23 |
| CHANNEL 1 - | 2 | CHANNEL 11 + | 24 |
| GROUND | 3 | CHANNEL 11 - | 25 |
| CHANNEL 4 + | 4 | CHANNEL 14 + | 26 |
| CHANNEL 4 - | 5 | CHANNEL 14 - | 27 |
| CHANNEL 7 + | 6 | GROUND | 28 |
| CHANNEL 7 - | 7 | CHANNEL 16 + | 29 |
| GROUND | 8 | CHANNEL 16 - | 30 |
| CHANNEL 10 + | 9 | CHANNEL 3 + | 31 |
| CHANNEL 10 - | 10 | CHANNEL 3 - | 32 |
| CHANNEL 13 + | 11 | GROUND | 33 |
| CHANNEL 13 - | 12 | CHANNEL 6 + | 34 |
| GROUND | 13 | CHANNEL 6 - | 35 |
| TRIGGER INPUT | 14 | CHANNEL 9 + | 36 |
| GROUND | 15 | CHANNEL 9 - | 37 |
| CHANNEL 2 + | 16 | GROUND | 38 |
| CHANNEL 2 - | 17 | CHANNEL 12 + | 39 |
| GROUND | 18 | CHANNEL 12 - | 40 |
| CHANNEL 5 + | 19 | CHANNEL 15 + | 41 |
| CHANNEL 5 - | 20 | CHANNEL 15 - | 42 |
| CHANNEL 8 + | 21 | GROUND | 43 |
| CHANNEL 8 - | 22 | GROUND | 44 |

The mating connector to J200, J201 or J202 is available from the following companies:

**ITT Cannon**

| | |
|---|---|
| P/N ZDBA44P | Connector |
| P/N ZD110238-1009 | Connector Pin |
| P/N 995-2000-022 | Crimp Tool |
| P/N 980-0004-804 | Insertion/Extraction Tool |

**Positronic Industries, Inc.**

| | |
|---|---|
| P/N ODD44M1FY0C | Connector |
| P/N 9507 | Crimp Tool |
| P/N 9502-4 | Contact Positioner |
| P/N M81969/1-04 | Insertion/Extraction Tool |

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

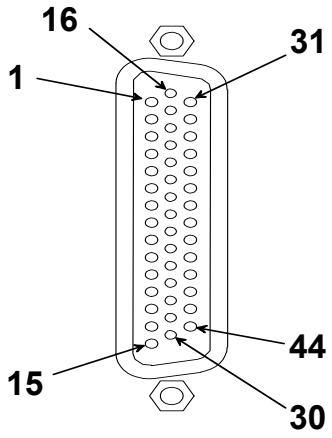



**FIGURE 2-2 J200, J201 AND J202 PIN LOCATIONS**

# SECTION 3

# INTRODUCTION

## EXAMPLES OF SCPI COMMANDS

### ABORT

The ABORt command takes the VM2716A scanning voltmeter out of the waiting-for-trigger mode.

**ABORt** — ***There are no command parameters.***

**EXAMPLES**

`ABORt` — *Takes the VM2716A scanning voltmeter out of the waiting-for-trigger mode.*

**CALIBRATION:COUNT?**

The CALibration:COUNt query returns a number that indicates the number of times the VM2716A 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 0. There are no query parameters.***

## EXAMPLES

```
CALibration:COUNt?
3
```

*Indicates the that the VM2716A has been calibrated 3 times.*

### 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

| | |
|---|---|
| `CALibration:DEFault` | *Sets the calibration gain and offset values to zero.* |
| `CALibration 1:GAIN?`<br>`0` | *Returns the default gain for Channel 1.* |

**CALIBRATION:GAIN**

The Calibration:GAIN command is used to set the calibration constant for the selected channel's gain and its effect is immediate. Note that the calibration security should be disabled for the calibration gain command to function, otherwise an error is generated.

| | |
|---|---|
| **CALibration<adjust>:GAIN<value>** | ***Where <adjust> is 1-2 referring to a specific calibration DAC:***<br><br>***1 = Channels 1 through 16 coarse***<br>***2 = Channels 1 through 16 fine***<br><br>***Where <value> ranges from -128 to +127.*** |

## EXAMPLES

| | |
|---|---|
| `CALibration 2:GAIN 100` | *Sets the fine value to 100* |
| `CALibration 2:GAIN?`<br>`100` | *Returns the gain value 100 for the fine adjust DAC.* |

### CALIBRATION:RESET

The CALIbration:RESet command restores the Calibration DAC values from the non-volatile memory.

| | |
|---|---|
| **CALibration:RESet** | ***There are no command parameters.*** |

## EXAMPLES

| | |
|---|---|
| `CALibration:RESet` | *Restores the Calibration DAC values from the non-volatile memory.* |

**CALIBRATION:SECURE:CODE**

The CALibration:SECure:CODE command sets the code required to disable the calibration security. The calibration security must be disabled in order to change the code string. The default code set by factory for VM2716A is 'VM2716A'. Note: The code is case sensitive.

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

## EXAMPLES

| | |
|---|---|
| `CALibration:SECure:STATe OFF,#17VM2716A` | *Sets the security OFF so that the calibration constants can be stored in the non-volatile memory. The password here is assumed to be VM2716A.* |
| `CALibration:SECure:CODE #15OLIVE` | *Sets the security code for OLIVE in IEEE-488.2 definite block format. Note that the password is case sensitive.* |
| `CALibration:SECure:STATe ON` | *Sets the security state ON so that the calibration cannot be stored in the non-volatile memory.* |

**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 state must be OFF or disabled. In order to disable the security state, the security code must be supplied and must be in 4 part block format. The 4 parts are:

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

| | |
|---|---|
| **CALibration:SECure:STATe,mode[,<code>]** | ***Where <code> is 0 \| OFF \| 1 \| ON.. OFF or 0 means values may be stored in the non-volatile memory. ON or 1 means values may not be stored in the non-volatile memory.*** <br><br> ***Where <code> is the parameter that must be present to disable the security, which comprises of 4 parts as described above. Note: The code is case sensitive.*** |

## EXAMPLES

| | |
|---|---|
| `CALibration:SECure:STATe OFF,#17VM2716A` | *Sets the security OFF so that the calibration constants can be stored in the non-volatile memory. The password here is assumed to be VM2716A. Note that the password is case sensitive.* |
| `CALibration:SECure:STATe?` <br> `OFF` | *Reports the state of the security which is currently set to OFF.* |
| `CALibration:SECure:STATe ON` | *Sets the security state ON so that the calibration cannot be stored in the non-volatile memory.* |
| `CALibration:SECure:STATe?` <br> `ON` | *Reports the state of the security which is currently set to ON.* |

**CALIBRATION:STORE**

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

| | |
|---|---|
| **CALibration:STORe** | ***The security state should be OFF prior to the usage of this command*** |

**EXAMPLES**

| | |
|---|---|
| `CAL1:GAIN -5` | *Programming the gain to -5.* |
| `CAL1:ZERO 2` | *Programming the offset to 2.* |
| `CALibration:STORe` | *Saves the current calibration constants to the non-volatile memory [Assume that the security state is OFF].* |

**CALIBRATION:ZERO**

The CALibration:ZERO command is used to set the calibration constant for the selected channel's offset and its effect is immediate. The CALibration:ZERO command will function only when the calibration security is disabled, otherwise an error is generated.

| | |
|---|---|
| **CALibration<adjust>:ZERO <value>** | ***Where <adjust> is 1 - 2 referring to a specific Calibration DAC:***<br><br>***1 = Channels 1 through 16 coarse***<br>***2 = Channels 1 through 16 fine***<br><br>***Where <value> ranges from -128 to +127.*** |

**EXAMPLES**

| | |
|---|---|
| `CALibration 2:ZERO 100` | *Sets the fine offset value to 100.* |
| `CALibration 2:ZERO?` | *Returns the fine offset value of 100.* |

**CONFIGURE:[VOLTAGE:]APERTURE**

The CONFigure:APERture command sets the amount of time over which a reading is measured. This is useful in making the lowest noise measurement possible at a given scan rate.

| | |
|---|---|
| **CONFigure:[VOLTage:]APERture <time>** | ***Where <time> is 4.0e-4 to 1.67e-2 in seconds.*** |

**EXAMPLES**

| | |
|---|---|
| `CONFigure:APERture 1.67e-2` | *Sets the measurement aperture time as 1.67e-2 seconds.* |
| `CONFigure:APERture?` | *Returns the measurement aperture time that is currently set at 1.67e-2 seconds.* |

### CONFIGURE:VOLTAGE

The CONFigure:VOLTage command sets the desired voltage range of the scanning voltmeter and places the instrument in the waiting-for-trigger mode. Note that an INIT command followed by a trigger event must occur before the instrument will take the desired readings.

| | |
|---|---|
| **CONFigure:VOLTage<range>,<channel_list>** | ***Where <range> is 0.1 \| 1 \| 10 volts.*** |
| | ***Where <channel_list> is the standard channel list format from 1 through 16.*** |

**EXAMPLES**

| | |
|---|---|
| `CONFigure:VOLTage 1,(@1,2,4,16)` | *Sets the voltage range of 1 for the Channels 1, 2, 4 and 16.* |
| `CONFigure:VOLTage?` | *Returns the voltage ranges and the configured channel list.* |

**FETCH?**

The FETCh query retrieves the measurements stored in the scanning voltmeter.

| | |
|---|---|
| **FETCh?** | ***There are no query parameters.*** |

**EXAMPLES**

| | |
|---|---|
| `FETCh?`<br>`8.253456` | *The query returns the voltmeter's measurement, i.e. 8.253456 stored in the scanning voltmeter.* |

**Note**: *This query retrieves the voltmeter's readings taken as a result of an Initiate command followed by a trigger event and places them into the output buffer in the format set by the FORMat:VOLTage:DATA command.*

**FORMAT:VOLTAGE:DATA**

The FORMat:VOLTage:DATA command sets the format length for retrieving data from the voltmeter.

| | |
|---|---|
| **FORMat:VOLTage[:DATA]<length>** | ***Where <length> is 1 to 7 significant digits in ASCII.*** |

**EXAMPLES**

| | |
|---|---|
| `FORMat:VOLTage +2.1` | *Sets the format length as +2.1 for retrieving data from the voltmeter. The above gives 2 digits before and 1 digit after the decimal point.* |
| `FORMat:VOLTage?` | *Returns the format length for retrieving data from scanning voltmeter.* |

**INITIATE**

The INITiate command places the scanning voltmeter into the waiting for trigger state. Note that the *RST or ABORt removes the scanning voltmeter from the waiting for trigger state.

| | |
|---|---|
| **INITiate** | ***There are no command parameters.*** |

## EXAMPLES

| | |
|---|---|
| `INITiate` | *Places the VM2716A in the waiting for trigger state.* |

**MEASURE:VOLTAGE**

The measure voltage command sets the desired voltage range of the scanning voltmeter and takes a reading. It is important to note that the above operation is the equivalent of sending the Abort command followed by the Configure Voltage command, followed by the Initiate command, followed by the Trigger command, followed by the Fetch command.

| | |
|---|---|
| **MEASure:VOLTage<range>,<channel_list>** | ***Where <range> is 0.01 \| 1 \| 10 volts.*** |
| | ***Where <channel_list> is the standard channel list format from 1 through 16.*** |

**EXAMPLES**

| | |
|---|---|
| `MEASure:VOLTage 1,(@1,2,4,16)` | *Sets the desired voltage range for the Channels 1, 2, 4 and 16.* |
| `MEASure:VOLTage?`<br>`1,(@1,2,4,16)` | *Returns the voltage range of 1 and the configured channel list.* |

**OUTPUT:TTLTRIG**

The OUTPut:TTLTrig command sets the trigger line to be used after the voltmeter finishes a single or a group of readings.

| | |
|---|---|
| **OUTPut:TTLTrig <line>** | ***Where the TTL trigger <line> can be from 0 to 7. Default = 0.*** |

## EXAMPLES

| | |
|---|---|
| `OUTPut:TTLTrig 3` | *Sets the output TTL trigger line to 3.* |
| `OUTPut:TTLTrig?`<br>`3` | *Indicates that TTL trigger line three is designated as an output.* |

### OUTPUT:TTLTRIG:POLARITY

The OUTput:TTLTtrig:POLarity command sets the polarity of the voltmeter's completed signal. A polarity of normal says the TTLTrig line goes low on completion. A polarity of invert says the TTL Trigger line goes high on completion.

| | |
|---|---|
| **OUTPut:TTLTrig:POLarity <polarity>** | ***Where the polarity chosen is NONE, NORMal or INVerted. Default = NORMal.*** |

## EXAMPLES

| | |
|---|---|
| `OUTPut:TTLTrig:POLarity NORM` | *Sets the polarity to normal.* |
| `OUTP:TTLTrig:POLarity?`<br>`NORM` | *Returns that the voltmeter's completed signal is set to normal polarity.* |

**OUTPUT:TTLTRIG:STATE**

The OUTPut:TTLTrig:STATE command enables the routing of the voltmeter complete (when task accomplished) signal to the VXIbus TTL trigger Lines. It is important to note that only one TTL trigger Line may be used at a time and so selecting a given trigger Line will disable any previously enabled TTL trigger line.

| | |
|---|---|
| **OUTPut[:TTLTrig] <n>:STATe <mode>** | ***Where <n> is the 8 TTL trigger lines. <n> can be 0 \| 1 \| 2 \| 3 \| 4 \| 5 \| 6 \| 7.*** <br><br> ***Where <mode> is 0 \| OFF \| 1 \| ON.*** |

**EXAMPLES**

| | |
|---|---|
| `OUTPut:TTLTrig 5 ON` | *Enables the routing of the voltmeter complete signal to the VXIbus TTL trigger line 5.* |
| `OUTPut:TTLTrig?` <br> `5,1` | *Returns the TTLTrigger line 5 that was selected and the state of the routing that is currently ON.* |
| `OUTP:TTLT5 OFF` | *Disabling routing of voltmeter complete signal.* |
| `OUTP:TTLT?` <br> `5,0` | |

### OUTPUT:TTLTRIG:TYPE

The OUTPut:TTLTrig:TYPE command determines what type of completed signal is generated. The command single says a completion signal is sent after each voltage is measured. The command group says a completion signal is sent after all channels in the scan have been measured.

| | |
|---|---|
| **OUTPut:[TTLTrig:]TYPE <trig type>** | ***Selects the type of completed signal. Where <trig type> is SINGle or GRoup. Default = GRoup*** |

## EXAMPLES

| | |
|---|---|
| `OUTPut:TYPE GR` | *Generates a signal to more than 1 TTL trigger line.* |
| `OUTPut:TYPE?`<br>`GR` | *Indicates that a signal generated will be enerated on more than one TTL trigger line.* |

**SAMPLE:COUNT**

The SAMPle:COUNt command sets the number of readings to be taken with each trigger event.

**SAMPle:COUNT<number>** ***Where <number> ranges from 1 to 16383.***

## EXAMPLES

| | |
|---|---|
| `SAMPle:COUNT 85` | *Sets the number of readings to be taken with each trigger event as 85.* |
| `SAMPle:COUNT?`<br>`85` | *Returns the number of readings, 85, taken with each trigger event.* |

### SAMPLE:TIMER

The SAMPle:TIMer command sets the period of the sample timer. The sample timer is used to pace the rate at which the scanning voltmeter scans through its scan list.

| | |
|---|---|
| **SAMPle:TIMer<period>** | ***Where <period> ranges from 1.0e-3 to 5.12e-1*** |

## EXAMPLES

| | |
|---|---|
| `SAMPle:TIMer 1e-3` | *Sets the period of the sample timer as 0.001.* |
| `SAMPle:TIMer?`<br>`0.1` | *Returns the period of the sample timer which is currently set at 0.001.* |

**TRACE:DATA:FEED**

The TRACe:DATA:FEED command sets how data is retrieved from the voltmeter.

| | |
|---|---|
| **TRACe:[DATA:]FEED <feed type>** | ***Data (measurements) are retrieved from the voltmeter using REGister or word serial transfers. Default = OFF.*** |

## EXAMPLES

| | |
|---|---|
| `TRACe:FEED OFF` | *Sets the voltmeter to retrieve data using word serial transfers.* |
| `TRACe:FEED?`<br>`OFF` | *Indicates that data is being received via word serial commands.* |

### TRIGGER:IMMEDIATE

The trigger immediate will cause the scanning voltmeter to start taking readings when in the waiting-for-trigger mode as determined by the scan list and the sample count commands. This command is similar to the *TRG command.

| | |
|---|---|
| **TRIGger[:IMMediate]** | ***No command parameters.*** |

## EXAMPLES

| | |
|---|---|
| `TRIGger` | *Causes a trigger event to occur for the scanning voltmeter.* |

**TRIGGER:SEQUENCE:SLOPE**

The TRIGger:SEQuence:SLOPe command selects the active edge of the input trigger to the voltmeter. The voltmeter must first be initiated before a trigger command can be given.

| | |
|---|---|
| **TRIGger:[SEQuence:]SLOPe <slope>** | ***Sets the rising or falling edge of the signal as the triggering source. Where the slope chosen is POSitive or NEGative. Default = NEGative.*** |

## EXAMPLES

| | |
|---|---|
| `TRIGger:SLOP POS` | *Sets the slope of the trigger on the Rising edge of the signal.* |
| `TRIG:SLOPe?`<br>`POS` | *Indicates that the rising edge of a signal will be used as a triggering source.* |

### TRIGGER:SEQUENCE:SOURCE

The TRIGger:SEQuence:SOURce command selects the input triggering source to the voltmeter.

| | |
|---|---|
| **TRIGger:[SEQuence:]SOURce <source>** | ***Selects the origin of the triggering source. Where IMMediate, EXTernal and TTLT lines are available. Default = IMMediate.*** |

## EXAMPLES

| | |
|---|---|
| `TRIG:SOURce EXT` | *Selects an external triggering device.* |
| `TRIGger:SOURce?`<br>`EXT` | *Indicates that an external triggering source is being used.* |

# APPLICATION EXAMPLES

This section contains examples of using SCPI command strings for programming the VM2716A module. The code is functional and will contain a brief description of the operation.

**Example 1**

In this example, the VM2716A sets the coarse Calibration Gain and the coarse Calibration Offset constants for all the channels. Then the changes are stored in the non-volatile memory (if the calibration security is OFF). It also returns the number of times the non-volatile memory is updated.

| | |
|---|---|
| `CAL:COUN?`<br>`4` | *Returns the number of times the calibration constants have been updated. This value will vary.* |
| `CAL:SEC:STAT 0,#17VM2716A` | *Turns the calibration security off using the VM2716A code.* |
| `CAL1:GAIN 2` | *Sets the coarse calibration gain to 2.* |
| `CAL1:ZERO 2` | *Sets the coarse calibration offset to 2.* |
| `CAL:STOR` | *Stores the calibration gain and offset constants in the non-volatile memory.* |
| `CAL:COUN?`<br>`5` | *Indicates that the non-volatile memory has been updated 5 times (the previous CAL:STOR command incremented the count by 1).* |
| `CAL:SEC:STAT 1` | *Disables further stores to the non-volatile memory.* |

**Example 2**

In this example, the following sequence of commands configures the voltmeter and puts it in the wait-for-trigger state. It then triggers the VM2716A module into reading the input voltages for the specified channels and returns the same.

| | |
|---|---|
| `CONF:VOLT 10,(@1,2,4)` | *Sets the voltage range 10 for Channels 1, 2 and 4.* |
| `CONF:APER 1.67e-2` | *Sets the measurement aperture time as 1.67e-2 seconds.* |
| `SAMP:COUN 85` | *Sets the number of readings to be taken with each trigger event to 85.* |
| `SAMP:SOUR TIM` | *Sets the pacing source for the sample period as TIMer.* |
| `SAMP:TIM 1e-3` | *Sets the period of sample timer as 0.001.* |
| `INIT` | *Places the scanning voltmeter into the wait-for-trigger state.* |
| `TRIG` | *Causes a trigger event to occur for the scanning voltmeter.* |
| `*OPC?`<br>`1` | *Querying the *OPC bit to check if the voltmeter has completed its reading.* |
| `FETC?`<br>`8.253456EQ,4.32835E0,3.543253E0` | *Returns the measurements stored in the scanning voltmeter.* |

# REGISTER ACCESS EXAMPLES

The VM2716A module supports register access for high-speed data retrieval.

The register at offset 0x22 contains the voltage data to be read. The register at offset 0x20 contains a bit that says data is available in the data register.

In order to interpret the 16-bit value read from the register at offset 0x22 as its equivalent voltage, the following steps must be followed:

a) The register at offset 0x22 must be read.

b) The 16-bit value read from the register at offset 0x22 must be converted to its equivalent voltage value. This would be done by multiplying the 16-bit value by (0.2/32768) for the 0.1 volt range, (2/32768) for the 1 volt range and (20/32768) for the 10 volt range.

For example:

Assuming that the value read from register at offset 0x22 is 16384 and the voltage range is 1 volt, the equivalent voltage value is 16384 * (2/32768) = 1 volt.

# VXIPLUG&PLAY EXAMPLES

```
/*
 *                              APPLICATION FUNCTION
 *                                   --------------------
 */
/*****************************************************************************************
***
Function:          vtvm2716A_setupAndMeasureVolts
```

Formal Parameters

ViSession instrHndl
- A unique session handle to the instrument.

ViInt16 range
- This parameter specifies the voltage range that is to be configured for the specified channels.

| Valid Values | Interpretation |
|---|---|
| vtvm2716A_RANGE_1 | 0.1 Volt Range |
| vtvm2716A_RANGE_2 | 1.0 Volt Range |
| vtvm2716A_RANGE_3 | 10.0 Volt Range |

ViInt16 channelList[]
- This parameter specifies the channels which are to be setup and read.

Each channel number in the array has the valid range :
vtvm2716A_CHANNEL_MIN (1) to
vtvm2716A_CHANNEL_MAX (16).

ViInt16 numOfChannels
- This parameter specifies the number of channels in the channel list.

Valid Range:
vtvm2716A_NUM_CHANNELS_MIN (1) to
vtvm2716A_NUM_CHANNELS_MAX (42)

ViReal64 apertureTime
- This parameter specifies the measurement aperture time that is to be configured for the instrument.

Valid Range (in secs):

vtvm2716A_APERTURE_TIME_MIN (4.0E-4) to
vtvm2716A_APERTURE_TIME_MAX (1.67E-2)

ViInt16 countNumber

- This parameter specifies the number of readings to be taken with each trigger event.

Valid Range:

vtvm2716A_COUNT_NUMBER_MIN (1) to
vtvm2716A_COUNT_NUMBER_MAX (16383)

ViBoolean pacingSource

- This parameter specifies the pacing source for the sample period. This parameter is considered only when the specified `countNumber' is greater than 1.

| Valid Values | Interpretation |
|---|---|
| vtvm2716A_SOURCE_IMM | Immediate |
| vtvm2716A_SOURCE_TIMER | Timer |

ViReal64 timerPeriod

- This parameter specifies the sample timer period. This parameter is considered only when the `pacingSource' specifies a value of vtvm2716A_SOURCE_TIMER.

Valid Range (in secs):

vtvm2716A_TIME_PERIOD_MIN (1.0E-3) to
vtvm2716A_TIME_PERIOD_MAX (5.12E-1)

ViReal64 voltageArray[]

- This parameter returns the voltages read by the instrument for specified channels.

ViPInt32 numVoltagesRead

- This paramter returns the voltage measurements read for the specified channels.

Description This function configures the voltmeter based on the input parameters and puts it into a wait-for-trigger state. It then triggers the instrument into reading the input voltages for the specified channels and returns the same.

**********
***/

```
ViStatus _VI_FUNC vtvm2716A_setupAndMeasureVolts(ViSession instrHndl,
                                        ViInt16         range,
                                        ViInt16                 channelList[],
                                        ViInt16                 numOfChannels,
                                        ViReal64                apertureTime,
                                        ViInt16                 countNumber,
                                        ViBoolean               pacingSource,
                                        ViReal64                timerPeriod,
                                ViReal64                voltageArray[],
                                ViPInt32                numVoltagesRead)

        {
                /* Variable used to store the return status of the function */
                ViStatus status = VI_NULL;

                /* Variable used to store the Operation Complete Query result */
                ViInt16 operComplete = VI_NULL;

                /* Variable used to store the query return from the instrument */
                ViChar  readBuf[vtvm2716A_READ_BUFF_SIZE];

                /*
                 * Configuring the Voltage range and aperture time for the
                 * specified channels
                 */
                status = vtvm2716A_configVoltParams(instrHndl, range, channelList,
                                        numOfChannels, apertureTime);
                        if (status < VI_SUCCESS)
                                return vtvm2716A_ERROR_CONFIGURING_VOLT_PARAMS;
```

```
/*
 * Configuring the Sample Count, the Pacing Source & the Pacing
 * Timer of the module
 */

status = vtvm2716A_configSampleParams (instrHndl, countNumber,
                                       pacingSource, timerPeriod);
            if (status < VI_SUCCESS)
                        return vtvm2716A_ERROR_CONFIGURING_SAMPLE_PARAMS;


/*
 * Setting the OUTP TTLT state off and configuring the number of
 * triggers to occur before the instrument returns to its idle
 * state
 */

status = vtvm2716A_configTrigParams(instrHndl, vtvm2716A_OUTPUT_TTLT1,
                                    vtvm2716A_MODE_OFF);
            if (status < VI_SUCCESS)
                        return vtvm2716A_ERROR_CONFIGURING_TRIGGER_PARAMS;


/*
 * Setting the instrument in Wait-for-Trigger state
 */

status = vtvm2716A_startStopInstr(instrHndl, vtvm2716A_INIT);
            if (status < VI_SUCCESS)
                        return vtvm2716A_ERROR_ARMING_THE_INSTRUMENT;

/*
 * Triggering the instrument
 */

status = vtvm2716A_triggerImmediate(instrHndl);
            if (status < VI_SUCCESS)
                        return vtvm2716A_ERROR_TRIGGERING_THE_INSTRUMENT;
```

```
	while(operComplete != 1)

	{
		/*
		 * Querying the OPC bit to check if the voltmeter has
		 * completed its readings
		 */

		status = vtvm2716A_sendWSCmd(instrHndl, "*OPC?", strlen("*OPC?"));
			if (status < VI_SUCCESS)
				return status;

		status = vtvm2716A_readInstrBuff(instrHndl, readBuf, vtvm2716A_READ_BUFF_SIZE);
			if (status < VI_SUCCESS)
				return status;

		status = vtvm2716A_checkInstrError(instrHndl);
			if (status < VI_SUCCESS)
				return status;

		operComplete = atoi(readBuf);
	}

	/*
	 * Reading the measured voltages.
	 */

	status = vtvm2716A_readVoltages(instrHndl, voltageArray, numVoltagesRead);
		if (status < VI_SUCCESS)
			return vtvm2716A_ERROR_READING_VOLTAGES;

	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 VM2716A 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 reset 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 VM2716A 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 Reset Value column gives the value of each command's setting when the unit is powered up or when a *RST command is executed.

## TABLE 4-1 IEEE 488.2 COMMON COMMANDS

| Command | Description | RST | Reset Value |
|---|---|---|---|
| CLS | Clears the Status Register. | | |
| ESE | Sets the Event Status Enable Register. | X | |
| ESR? | Queries and clears the Standard Event Status Register. | | |
| IDN? | Query the module Identification string. | | |
| OPC | Set the OPC bit in the Event Status Register. | | |
| RST | Resets the module to a known state. | | |
| STB? | Query the Status Byte Register. | | |
| TRG | Causes a trigger event to occur. | | |
| TST? | Starts and reports a self-test procedure. | | |
| WAI | Halts execution and queries. | X | |

## TABLE 4-2 INSTRUMENT SPECIFIC SCPI COMMANDS

| Command | Description | RST | Reset Value |
|---|---|---|---|
| ABORt | Takes the VM2716A scanning voltmeter out of the waiting-for-trigger mode. | | |
| CALibration:COUNt? | Returns a number that indicates the number of times the VM2716A has been calibrated. | | |
| CALibration:DEFault | Sets calibration gain and zero values to defaults. | | |
| CALibration:GAIN | Used to set the calibration gain constant. | | |
| CALibration:RESet | Restores the Calibration DAC values from the non-volatile memory. | | |
| CALibration:SECure:CODE | Sets the code required to disable calibration security. | | |
| CALibration[:SECure]:STATe | Enables and disables calibration security. | | |
| CALibration:STORe | Saves the current calibration constants into non-volatile memory. | | |
| CALibration:ZERO | Used to set the calibration offset constant. | | |
| CONFigure:[VOLTage:]APERture | Sets the measurement aperture time. | X | 400e-6 |
| CONFigure:VOLTage | Sets the voltage range for the scanning voltmeter. | X | 10 |
| DOWNload | Loads firmware updates into module. | | |
| FETCh? | Retrieves measurements stored in the scanning voltmeter. | | |
| FORMat:VOLTage[:DATA] | Sets the data format length for retrieving data from the voltmeter. | X | +2.2 |
| INITiate | Places the scanning voltmeter into the waiting-for-trigger state. | | |
| IT | Perform a combined initiate and trigger. | | |
| J | Short command to trigger. | | |
| MEASure[:VOLTage] | Sets the voltage range for the scanning voltmeter and takes a reading. | X | 10 |
| OUTPut:TTLTrig | Sets which line to use. | | |
| OUTPut:TTLTrig:POLarity | Determines the polarity of the voltmeter completed signal to the VXIbus trigger lines. | X | NORMal |
| OUTPut[:TTLTrig]:STATe | Enables routing the voltmeter complete signal to the VXIbus TTL trigger lines. | X | OFF |
| OUTPut:TTLTrig:TYPE | Determines what type of voltmeter completion signal is generated. | X | GRoup |
| SAMPle:COUNt | Sets the number of readings taken with each trigger event. | X | 16 |
| SAMPle:TIMer | Sets the sample timer period. | X | 1.0e-3 |
| TRACe[:DATA]:FEED | Sets how data is retrieved from the voltmeter. | | |
| TRIGger:[IMMediate] | Causes a trigger event to occur for the scanning voltmeter. | | |
| TRIGger[:SEQuence]:SLOPe | Sets the active edge of the voltmeter trigger. | X | NEGative |
| TRIGger[:SEQuence]:SOURce | Sets the source of the voltmeter trigger. | X | IMMediate |
| TRIGger:TYPE | Sets how the module operates after a trigger. | X | ONCE |

## TABLE 4-3 SCPI REQUIRED COMMANDS

| Command | Description | RST | Reset Value |
|---|---|---|---|
| STATus:OPERation:CONDition? | Queries the Operation Status Condition Register. | | |
| STATus:OPERation:ENABle? | Sets the Operation Status Enable Register. | X | 0 |
| STATus:OPERation[:EVENt]? | Queries the Operation Status Event Register. | | |
| STATus:PRESet | Presets the Status Register. | | |
| STATus:QUEStionable: CONDition? | Queries the Questionable Status Condition Register. | | |
| STATus:QUEStionable:ENABle? | Sets the Questionable Status Enable Register. | X | |
| STATus:QUEStionable[:EVENt]? | Queries the Questionable Status Event Register | | |
| SYSTem:ERRor? | Queries the Error Queue. | | 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 command such as an event or setting. |
| **Command Syntax** | Details the exact command format. |
| **Command Parameters** | Describes the parameters sent with the command and their legal range. |
| **Reset Value** | Describes the values assumed when the *RST 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** | Present 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. |

# COMMON SCPI COMMANDS

## *CLS

| | | |
|---|---|---|
| **Purpose** | Clears all status and event registers | |
| **Type** | IEEE 488.2 Common Command | |
| **Command Syntax** | *CLS | |
| **Command Parameters** | N/A | |
| ***RST Value** | N/A | |
| **Query Syntax** | N/A | |
| **Query Parameters** | N/A | |
| **Query Response** | N/A | |
| **Description** | This command clears the Status Event Register, Operation Status Register and the Questionable Data/Signal Register. It also clears the OPC flag and clears all queues (except the output queue). | |
| **Examples** | **Command / Query** | **Response / Descriptions** |
| | `*CLS` | *(Clears all status and event registers)* |
| **Related Commands** | N/A | |

# *ESE

| | | |
|---|---|---|
| **Purpose** | Sets the bits of the Event Status Enable Register | |
| **Type** | IEEE 488.2 Common Command | |
| **Command Syntax** | *ESE <mask> | |
| **Command Parameters** | <mask> = numeric ASCII value | |
| ***RST Value** | N/A, the parameter is required | |
| **Query Syntax** | *ESE? | |
| **Query Parameters** | N/A | |
| **Query Response** | Numeric ASCII value from 0 to 255 | |
| **Description** | The Event Status Enable (ESE) 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 Event Status Bit (ESB) in the Status Byte by the corresponding bit in the Event Status Register (ESR). If the ESB is set in the Service Request Enable (SRE) register, then an interrupt will be generated. See the *ESR? command for details regarding the individual bits. The ESE register layout is:<br><br>Bit 0 - Operation Complete<br>Bit 1 - Request Control<br>Bit 2 - Query Error<br>Bit 3 - Device Dependent Error<br>Bit 4 - Execution Error<br>Bit 5 - Command Error<br>Bit 6 - User Request<br>Bit 7 - Power On<br><br>The Event Status Enable query reports the current contents of the Event Status Enable Register. | |
| **Examples** | **Command / Query** | **Response *(Description)*** |
| | `*ESE 36`<br>`*ESE?` | 36 *(Returns the value of the event status enable register)* |
| **Related Commands** | *ESR? | |

# *ESR?

| | | |
|---|---|---|
| **Purpose** | Queries and clears the Standard Event Status Register | |
| **Type** | IEEE 488.2 Common Command | |
| **Command Syntax** | N/A | |
| **Command Parameters** | N/A | |
| ***RST Value** | N/A | |
| **Query Syntax** | ESR? | |
| **Query Parameters** | N/A | |
| **Query Response** | Numeric ASCII value from 0 to 255 | |
| **Description** | The Event Status Register (ESR) 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 Event Status Bit (ESB) in the Status Byte. The layout of the ESR is:<br><br>Bit 0 - Operation Complete<br>Bit 1 - Request Control<br>Bit 2 - Query Error<br>Bit 3 - Device Dependent Error<br>Bit 4 - Execution Error<br>Bit 5 - Command Error<br>Bit 6 - User Request<br>Bit 7 - Power On<br><br>The Operation Complete bit is set by the VM2716A when it receives an *OPC command.<br><br>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.<br><br>The Execution Error bit is set when an execution error is detected. Errors that range from -200 to -299 are execution errors.<br><br>The Command Error bit is set when a command error is detected. Errors that range from -100 to -199 are command errors.<br><br>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** | Queries the module for its identification string | |
| **Type** | IEEE 488.2 Common Command | |
| **Command Syntax** | N/A | |
| **Command Parameters** | N/A | |
| ***RST Value** | N/A | |
| **Query Syntax** | *IDN? | |
| **Query Parameters** | N/A | |
| **Query Response** | ASCII character string | |
| **Description** | The Identification (IDN) query returns the identification string of the 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.,VM2716A,0,1.xx<br>*(The revision listed here is for reference only; the response will always be the current revision of the instrument.)* |
| **Related Commands** | N/A | |

# *OPC

| | | |
|---|---|---|
| **Purpose** | Sets the OPC bit in the Event Status Register | |
| **Type** | IEEE 488.2 Common Command | |
| **Command Syntax** | *OPC | |
| **Command Parameters** | N/A | |
| ***RST Value** | N/A | |
| **Query Syntax** | *OPC? | |
| **Query Parameters** | N/A | |
| **Query Response** | 1 | |
| **Description** | The Operation Complete (OPC) command sets the OPC bit in the Event Status Register when all pending operations have completed. The OPC query will return a 1 to the output queue when all pending operations have completed. | |
| **Examples** | **Command / Query** | **Response *(Description)*** |
| | `*OPC`<br>`*OPC?` | *(Sets the OPC bit in the Event Status Register)*<br>1 *(Returns the value of the Event Status Register)* |
| **Related Commands** | *WAI | |

# *RST

| | | |
|---|---|---|
| **Purpose** | Resets the module's hardware and software to a known state | |
| **Type** | IEEE 488.2 Common Command | |
| **Command Syntax** | *RST | |
| **Command Parameters** | N/A | |
| ***RST Value** | N/A | |
| **Query Syntax** | N/A | |
| **Query Parameters** | N/A | |
| **Query Response** | N/A | |
| **Description** | The Reset (RST) command resets the module's hardware and software to a known state. See the command index at the beginning of this chapter for the default parameter values used with this command. | |
| **Examples** | **Command / Query** | **Response *(Description)*** |
| | *RST | *(Resets the module)* |
| **Related Commands** | N/A | |

# *STB?

| | | |
|---|---|---|
| **Purpose** | Queries the Status Byte Register | |
| **Type** | IEEE 488.2 Common Command | |
| **Command Syntax** | N/A | |
| **Command Parameters** | N/A | |
| ***RST Value** | N/A | |
| **Query Syntax** | *STB? | |
| **Query Parameters** | N/A | |
| **Query Response** | Numeric ASCII value from 0 to 255 | |
| **Description** | The Read Status Byte (STB) 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:<br><br>Bit 0 – Unused<br>Bit 1 – Unused<br>Bit 2 – Error Queue Has Data<br>Bit 4 – Questionable Status Summary (not used)<br>Bit 5 – Message Available<br>Bit 6 – Master Summary Status<br>Bit 7 – Operation Status Summary | |
| **Examples** | **Command / Query** | **Response *(Description)*** |
| | *STB? | 20 (*Queries the Status Byte Register)* |
| **Related Commands** | N/A | |

# *TRG

| | | |
|---|---|---|
| **Purpose** | Causes a trigger event to occur | |
| **Type** | IEEE 488.2 Common Command | |
| **Command Syntax** | *TRG | |
| **Command Parameters** | N/A | |
| ***RST Value** | N/A | |
| **Query Syntax** | N/A | |
| **Query Parameters** | N/A | |
| **Query Response** | N/A | |
| **Description** | This command generates a short pusle or a word serial command.<br><br>The Trigger command causes a trigger event to occur. | |
| **Examples** | **Command / Query** | **Response *(Description)*** |
| | `*TRG` | *(Triggers an event)* |
| **Related Commands** | TRIGger:SEQuence:IMMediate | |

# *TST?

| | | |
|---|---|---|
| **Purpose** | Causes a self-test procedure to occur and queries the results | |
| **Type** | IEEE 488.2 Common Command | |
| **Command Syntax** | N/A | |
| **Command Parameters** | N/A | |
| ***RST Value** | N/A | |
| **Query Syntax** | *TST? | |
| **Query Parameters** | N/A | |
| **Query Response** | Numeric ASCII value from 0 to 143 | |
| **Description** | The Self-Test query causes the VM2716A to run its self-test procedures and report on the results. A non-zero value indicates an error condition. | |
| **Examples** | **Command / Query** | **Response *(Description)*** |
| | *TST | 0 (*Begins the self-test procedure returns the result)* |
| **Related Commands** | N/A | |

# *WAI

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

# INSTRUMENT SPECIFIC SCPI COMMANDS

## ABORt

| | | |
|---|---|---|
| **Purpose** | Takes the VM2716A scanning voltmeter out of the waiting-for-trigger mode | |
| **Type** | Event | |
| **Command Syntax** | ABORt | |
| **Command Parameters** | None | |
| ***RST Value** | N/A | |
| **Query Syntax** | None | |
| **Query Parameters** | N/A | |
| **Query Response** | N/A | |
| **Description** | The Abort command takes the VM2716A scanning voltmeter out of the waiting-for-trigger mode. | |
| **Examples** | **Command / Query** | **Response *(Description)*** |
| | ABOR | |
| **Related Commands** | INITiate | |

# CALibration:COUNt?

| | | |
|---|---|---|
| **Purpose** | Returns a number that indicates the number of times the VM2716A has been calibrated | |
| **Type** | Query | |
| **Command Syntax** | None - Query Only | |
| **Command Parameters** | N/A | |
| ***RST Value** | N/A | |
| **Query Syntax** | CALibration:COUNt? | |
| **Query Parameters** | N/A | |
| **Query Response** | N/A | |
| **Description** | The instrument will increment the count every time the non-volatile memory storing the calibration constants is updated. If the calibration security is disabled (CALibration:SECure:STATe OFF set) and CALibration:STORe:AUTO ON is set, the count will increment with each execution of the CALibration:GAIN or CALibration:ZERO command. If the CALibration:STORe:AUTO OFF is set, the count will only be incremented by invoking the CALibration:STORe command.<br><br> **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:COUNT?` | 5 |
| **Related Commands** | CALibration:STORe | |

# CALibration:DEFault

| | | |
|---|---|---|
| **Purpose** | Sets calibration gain and zero values to defaults | |
| **Type** | Event | |
| **Command Syntax** | CALibration:DEFault | |
| **Command Parameters** | N/A | |
| ***RST Value** | N/A | |
| **Query Syntax** | N/A | |
| **Query Parameters** | N/A | |
| **Query Response** | N/A | |
| **Description** | Sets all CAL:ZERO and CAL:GAIN values to zero.<br> **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 <adjust>:GAIN value<br>CALibration <adjust>:ZERO value | |

# CALibration:GAIN

| | |
|---|---|
| **Purpose** | Used to set the calibration gain constant |
| **Type** | Event |
| **Command Syntax** | CALibration<adjust>:GAIN <value> |
| **Command Parameters** | <adjust> = 1 – 2 referring to a specific Calibration DAC:<br>1 = Channels 1 through 16 coarse<br>2 = Channels 1 through 16 fine<br><value> = The range for value is -128 to +127. The query returns the value from the non-volatile memory rather than the currently used value and may be different from the constant currently being used. |
| ***RST Value** | N/A |
| **Query Syntax** | CALibration<adjust>:GAIN? |
| **Query Parameters** | <adjust> = 1 - 2 referring to a specific Calibration DAC:<br>1 = Channels 1 through 16 coarse<br>2 = Channels 1 through 16 fine |
| **Query Response** | -128 to +127 in the specified format |
| **Description** | The Calibration Gain command sets the gain constant for all the channels; the effect is immediate. If the CALibration:STORe:AUTO ON is set, the command will save the new constant to the non-volatile memory each time the command is set. If the CALibration:STORe:AUTO OFF is set, a CALibration:STORe command must be executed in order to save the new constant. The Calibration Gain command will only function when calibration security is disabled, otherwise an error is generated.<br> **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)*** |
|---|---|---|
| | `CAL2:GAIN -120`<br>`CAL2:GAIN?` | <br>-120 |
| **Related Commands** | CALibration<adjust>:ZERO<value> | |

# CALibration:RESet

| | | |
|---|---|---|
| **Purpose** | Restores Calibration DAC values from non-volatile memory | |
| **Type** | Event | |
| **Command Syntax** | CALibration:RESet | |
| **Command Parameters** | N/A | |
| ***RST Value** | N/A | |
| **Query Syntax** | N/A | |
| **Query Parameters** | N/A | |
| **Query Response** | N/A | |
| **Description** | This command will load the Calibration DACs from the calibration constants stored in non-volatile memory. This allows the user to modify Calibration DAC values with the ability to restore the original values.<br><br> **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 | |
| **Related Commands** | None | |

# CALibration:SECure:CODE

| | |
|---|---|
| **Purpose** | Sets the code required to disable calibration security |
| **Type** | Event |
| **Command Syntax** | CALibration:[SECure]:CODE <code> |
| **Command Parameters** | <code> = The code string can be from 1 to 12 ASCII characters in length entered in IEEE-488.2 definite or indefinite length arbitrary block format |
| ***RST Value** | N/A |
| **Query Syntax** | None |
| **Query Parameters** | N/A |
| **Query Response** | N/A |
| **Description** | Calibration security must have been previously disabled in order to change the code string. Before shipping the instrument, the factory sets the code to 'VM2716A'.<br> **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:CODE OFF,#17VM2716A` | *(Sets the security OFF so that the calibration constants can be stored in the non-volatile memory. The password here is assumed to be VM2716A.)* |
| | `CAL:CODE #15OLIVE` | *(Sets the security code for OLIVE in IEEE-488.2 definite block format. Note that the password is case sensitive)* |
| | `CAL:SEC:CODE ON` | *(Sets the security state ON so that the calibration cannot be stored in the non-volatile memory.)* |
| **Related Commands** | CALibration:SECure:STATe<mode>[,<code>] | |

# CALibration[:SECure]:STATe

| | |
|---|---|
| **Purpose** | Enables and disables calibration security |
| **Type** | Event |
| **Command Syntax** | CALibration:[SECure]:STATe <mode>[,<code>] |
| **Command Parameters** | <mode> = 0 \| 1 \| ON \| OFF<br><code> = The code parameter must be present to disable the security or it generates error -109, "Missing parameter". The value must match the currently programmed security code or it generates error -224, "Illegal parameter value". To enable security, the code parameter is not required, but if it is provided, it will be checked. If the code is given but does not match the current security code, error -224, "Illegal parameter value" will be generated. |
| ***RST Value** | N/A |
| **Query Syntax** | CALibration:[SECure]:STATe? |
| **Query Parameters** | None |
| **Query Response** | 1 \| 0 |
| **Description** | The module powers up with the secure state on. While security is on, no stores to non-volatile memory are allowed. This command turns the state on or off. In order to turn the state to off, the current security code must be supplied. To turn on security, code does not need to be supplied. If it is supplied the code is checked. The security code must be supplied in IEEE-488.2 definite or indefinite length arbitrary block format.<br><br>**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:STAT OFF,#17VM2716A`<br>`CAL:STAT ON` | |

| | |
|---|---|
| **Related Commands** | CALibration:SECure:CODE |

# CALibration:STORe

| | | |
|---|---|---|
| **Purpose** | Saves the current calibration constants into 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:SECure:STATe must be OFF before using this command.<br> **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:STOR` | |
| **Related Commands** | CALibration<adjust>:GAIN<value><br>CALibration<adjust>:ZERO<value> | |

# CALibration:ZERO

| | |
|---|---|
| **Purpose** | Used to set the calibration offset constant |
| **Type** | Event |
| **Command Syntax** | CALibration<adjust>:ZERO <value> |
| **Command Parameters** | <adjust> = 1 \| 2, referring to a specific Calibration DAC:<br>1 = Channels 1 through 16 coarse<br>2 = Channels 1 through 16 fine<br><value> = The range for value is -128 to +127. The query returns the value from the non-volatile memory rather than the currently used value and may be different from the constant currently being used. |
| ***RST Value** | N/A |
| **Query Syntax** | CALibration<adjust>:ZERO? |
| **Query Parameters** | <adjust> = 1 \| 2, referring to a specific Calibration DAC:<br>1 = Channels 1 through 16 coarse<br>2 = Channels 1 through 16 fine |
| **Query Response** | -128 to +127 in the specified format |
| **Description** | The Calibration Zero command sets the offset constant for all the channels; the effect is immediate. If the CALibration:STORe:AUTO ON is set, the command will save the new constant to the non-volatile memory each time the command is set. If the CALibration:STORe:AUTO OFF is set, a CALibration:STORe command must be executed in order to save the new constant. The Calibration Zero command will only function when calibration security is disabled, otherwise an error is generated.<br> **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:ZERO 115`<br>`CAL1:ZERO?` | <br>115 |
| **Related Commands** | CALibration<adjust>:GAIN<value> | |

# CONFigure:[VOLTage:]APERture

| | | |
|---|---|---|
| **Purpose** | Sets the measurement aperture time | |
| **Type** | Setting | |
| **Command Syntax** | CONFigure:[VOLTage:]APERture <time> | |
| **Command Parameters** | <time> = 4.0e-4 to 0.2 in seconds | |
| ***RST Value** | 400e-6 | |
| **Query Syntax** | CONFigure:[VOLTage:]APERture? | |
| **Query Parameters** | None. | |
| **Query Response** | 4.0e-4 through 2.00e-1 | |
| **Description** | The Configure Aperture command sets the amount of time over which a reading is measured. This is useful in making the lowest noise measurement possible at a given scan rate. For example when taking accurate voltage measurements in a 60 Hz environment, an aperture of 16.7 ms will reject the 60 Hz component of the measured DC voltage. A setting of 20 ms will reject a 50 Hz component. For high speed scans, a 400 $\mu$s aperture should be used. | |
| **Examples** | **Command / Query** | **Response *(Description)*** |
| | `CONF:APER 1.67e-2` | |
| **Related Commands** | None | |

# CONFigure:VOLTage

| | |
|---|---|
| **Purpose** | Sets the voltage range for the scanning voltmeter and sets the channel list |
| **Type** | Setting |
| **Command Syntax** | CONFigure:VOLTage <range>,<channel_list> |
| **Command Parameters** | <range> = 0.1 \| 1 \| 10<br><channel_list> = (@1) through (@16) for single channel readings<br><br>(@begin_channel:end_channel) for all channels between begin_channel and end_channel. Begin_channel is a lower channel number than end_channel.<br><br>(@first_channel, second_channel, ... , last_channel) for sequential channels.<br><br>See the *SCPI Syntax and Style* manual for more information on channel lists. |
| ***RST Value** | 10,(@1:16) |
| **Query Syntax** | CONFigure:VOLTage? |
| **Query Parameters** | None |
| **Query Response** | 0.1 \| 1 \| 10, <channel_list> |
| **Description** | The Configure Voltage command sets the desired voltage range of the scanning voltmeter. An INIT command followed by a trigger event must occur before the instrument will take the desired readings. This command also sets the number of readings to take for each trigger. The number of readings can be changed with the SAMPle:COUNt command.<br><br>Each voltage range supports a 60% over-range reading, therefore, the 10 volt range can measure voltages from -16 volts to +16 volts, the 1 volt range can measure voltages from -1.6 volts to +1.6 volts and the 0.1 volt range can measure voltages from -0.16 volts to +0.16 volts<br><br>Channels can be repeated to read multiple times. For example, this is a valid list: (@1,2,1,3,1,4,1,5,1,6,1,7,1,8). The maximum number of channels in a list is 100. |

| **Examples** | **Command / Query** | **Response *(Description)*** |
|---|---|---|
| | `CONF:VOLT 1,(@1,2,3,16)` | |
| | `CONF:VOLT 0.1,(@1:8,12:16)` | |
| | `CONF:VOLT?` | 0.1,(@1:8,10,12:16) |

| | |
|---|---|
| **Related Commands** | MEASure:VOLTage<br>SAMPle:COUNt |

# DOWNLOAD

| | |
|---|---|
| **Purpose** | Loads firmware updates into a module |
| **Type** | Event |
| **Command Syntax** | DOWNLOAD <section> |
| **Command Parameters** | <section> = 0 \| 1 \| 2 \| 3<br>0 is for the VMIP base code<br>1 is for the module in the top position<br>2 is for the module in the middle position<br>3 is for the module in the bottom position |
| ***RST Value** | N/A |
| **Query Syntax** | N/A |
| **Query Parameters** | None |
| **Query Response** | N/A |
| **Description** | Firmware for VMIP cards is stored in FLASH memory which can be updated in the field. Each card has at least 4 separate areas where firmware is stored. This command is used to update one area of the firmware. To update a module, do the following:<br><br>1. Obtain the proper HEX file from the factory.<br>2. Send the DOWNLOAD command with the appropriate <section>.<br>3. Send each line of the HEX file to the module (one line at a time).<br>4. Wait at least 15 seconds after the last line of the HEX file is sent.<br>5. Turn power to the module off.<br>6. Turn power back on and perform normal start up operations.<br><br>If more than one module needs to be updated, repeat the procedure for each section to be updated.<br><br> **Improper use of this command can completely disable the whole card**.<br><br>**NOTE**: *This command can only be used on the module in the top position. If there is no module in the top position, there is a dummy application installed. In the case of no top module, the command is sent to the dummy application. The top module/application always has an address that is an integer multiple of 4. Even though the command and the HEX file are always sent to the top module/application, the module corresponding to the <section> is the one that is updated.* |

| **Examples** | **Command / Query** | **Response *(Description)*** |
|---|---|---|
| | `DOWNLOAD 1`<br>`S0...`<br>`:`<br>`S8...` | *(First line of HEX file)*<br><br>*(Last line of HEX file)* |
| **Related Commands** | None | |

# FETCh?

| | | |
|---|---|---|
| **Purpose** | Retrieves measurements stored in the scanning voltmeter | |
| **Type** | Event | |
| **Command Syntax** | FETCh? | |
| **Command Parameters** | None | |
| ***RST Value** | N/A | |
| **Query Syntax** | N/A | |
| **Query Parameters** | N/A | |
| **Query Response** | N/A | |
| **Description** | The Fetch command places the voltmeter's readings taken as a result of an INITiate command followed by a trigger event and places them into the output buffer in the format set by the FORMat:VOLTage:DATA command. | |
| **Examples** | **Command / Query** | **Response *(Description)*** |
| | FETC? | 8.253456E0 |
| **Related Commands** | READ? | |

# FORMat[:VOLTage][:DATA]

| | | |
|---|---|---|
| **Purpose** | Sets the data format length for retrieving data from the voltmeter | |
| **Type** | Setting | |
| **Command Syntax** | FORMat[:VOLTage][:DATA] <length> | |
| **Command Parameters** | <length> = X.Y, where X is the number of digits before the decimal and Y is the number of digits after the decimal. $1 \le X \le 3$; $0 \le Y \le 6$ | |
| ***RST Value** | 3.6 | |
| **Query Syntax** | FORMat:VOLTage[:DATA]? | |
| **Query Parameters** | None. | |
| **Query Response** | Returns the value currently set for the <length> parameter | |
| **Description** | The Format Data command sets the format length for retrieving data as follows:<br><br>ASCII - Data is transferred in the form. Each reading is followed by a comma(,). A line feed (LF) and End-Or-Identify (EOI) follow the last reading. | |
| 7 | **Command / Query** | **Response *(Description)*** |
| | `FORM +2.1`<br><br>`FORM?` | *(Gives two digits before and one digit after the decimal point)*<br><br>2.100000 |
| **Related Commands** | None | |

# INITiate

| | | |
|---|---|---|
| **Purpose** | Places the scanning voltmeter into the waiting-for-trigger state | |
| **Type** | Event | |
| **Command Syntax** | INITiate | |
| **Command Parameters** | None | |
| ***RST Value** | N/A | |
| **Query Syntax** | N/A | |
| **Query Parameters** | N/A | |
| **Query Response** | None | |
| **Description** | The INITiate command is used in conjunction with the TRIGger:SOURce command and the TRIGger:TYPE command to allow the voltmeter to take readings. If the trigger source is selected to be immediate, then a TRIGger command will trigger a reading when sent to the instrument. If a different trigger source is selected, then the trigger condition must be met after the INITiate command is sent to cause a measurement to be made.<br><br>*RST removes the scanning voltmeter from the waiting for trigger state. | |
| **Examples** | **Command / Query** | **Response *(Description)*** |
| | `INIT` | |
| **Related Commands** | TRIGger:SOURce<br>TRIGger:TYPE<br>CONFigure:VOLTage<br>IT | |

# IT

| | | |
|---|---|---|
| **Purpose** | Places the voltmeter in the waiting-for-trigger state and triggers it. | |
| **Type** | Event | |
| **Command Syntax** | IT | |
| **Command Parameters** | None | |
| ***RST Value** | N/A | |
| **Query Syntax** | N/A | |
| **Query Parameters** | N/A | |
| **Query Response** | None | |
| **Description** | This is not a SCPI command. It is a short command that performs two SCPI commands for the user. The commands are INITiate and TRIGger. The short command takes less time to transfer across the VXIbus and to be recognized by the module. | |
| **Examples** | **Command / Query** | **Response *(Description)*** |
| | `IT` | |
| **Related Commands** | INITiate<br>TRIGger[:IMMediate] | |

# J

| | | |
|---|---|---|
| **Purpose** | Causes a trigger event to occur for the voltmeter | |
| **Type** | Event | |
| **Command Syntax** | J | |
| **Command Parameters** | None | |
| ***RST Value** | N/A | |
| **Query Syntax** | N/A | |
| **Query Parameters** | N/A | |
| **Query Response** | None | |
| **Description** | This is not a SCPI command. It is a short command that performs one SCPI command for the user. That command is TRIGger. The short command takes less time to transfer across the VXIbus and to be recognized by the module. | |
| **Examples** | **Command / Query** | **Response *(Description)*** |
| | J | |
| **Related Commands** | TRIGger[:IMMediate] | |

# MEASure[:VOLTage]

| | |
|---|---|
| **Purpose** | Sets the voltage range for the scanning voltmeter, sets the channel list, and takes a reading. |
| **Type** | Setting |
| **Command Syntax** | MEASure[:VOLTage] <range>,<channel_list> |
| **Command Parameters** | <range> = 0.1 \| 1 \|10<br><channel_list> = (@1) through (@16) for single channel readings<br><br>(@begin_channel:end_channel) for all channels between begin_channel and end_channel. Begin_channel is a lower channel number than end_channel.<br><br>(@first_channel, second_channel, …, last_channel) for sequential channels.<br><br>See the *SCPI Syntax and Style* manual for more information. |
| ***RST Value** | 10,(@1:16) |
| **Query Syntax** | MEASure:[VOLTage]? |
| **Query Parameters** | None. |
| **Query Response** | 0.1 \| 1 \| 10, <channel_list> |
| **Description** | The Measure Voltage command sets the desired voltage range of the scanning voltmeter and takes a reading. It is the equivalent to sending the Abort command followed by the Initiate command followed by the Fetch command. This command also sets the number of readings to take for each trigger to the number of channels in the channel list.<br><br>Each voltage range supports a 60% over-range reading, therefore, the 10 volt range can measure voltages from -16 volts to +16 volts, the 1 volt range will read voltages from -1.6 volts to +1.6 volts and the 0.1 volt range will read voltages from -0.16 volts to +0.16 volts. |

| **Examples** | **Command / Query** | **Response *(Description)*** |
|---|---|---|
| | `MEAS 1,(@1,2,4,23)` | 0.1,0.2,0.3,0.4 *(Returns voltages from Channels 1, 2, 4 and 23)* |
| | `MEAS 0.1,(@1:24,28,30:32)` | 0.01,0.02, … *(28 values total returned)* |
| | `MEAS?` | 0.1,(@1:24,28,30:32) |
| **Related Commands** | CONFigure:VOLTage | |

# OUTput:TTLTrig

| | | |
|---|---|---|
| **Purpose** | Selects which line to use. | |
| **Type** | Setting | |
| **Command Syntax** | OUTPut:TTLTrig <line> | |
| **Command Parameters** | <line> = 0 \| 1 \| 2 \| 3 \| 4 \| 5 \| 6 \| 7 | |
| ***RST Value** | 0 | |
| **Query Syntax** | OUTput:TTLTrig? | |
| **Query Parameters** | None | |
| **Query Response** | | |
| **Description** | When the voltmeter finishes a single reading or a group of readings, it can generate a completed signal on one of the VXIbus TTL trigger lines. The trigger line used is set by the OUTPut:TTLTrig command. The output is enabled by the OUTPut:STATe command. The type of completed signal (completed a single reading or completed a group of readings) is set by the OUTPut:TYPE command. The OUTPut:POLarity command sets the polarity of the voltmeter complete signal.<br><br>A line of 0 selects TTL line 0. A line of 1 selects TTL line 1, etc. | |
| **Examples** | **Command / Query** | **Response *(Description)*** |
| | `OUTP:TTLT 3`<br>`OUTP:TTLT?` | <br>3 |
| **Related Commands** | OUTPut:STATe<br>OUTPut:TYPE<br>OUTPut:POLarity | |

# OUTPut:TTLTrig:POLarity

| | | |
|---|---|---|
| **Purpose** | Determines the polarity of the voltmeter completed signal to the VXIbus trigger lines | |
| **Type** | Setting | |
| **Command Syntax** | OUTPut[:TTLTrig]:POLarity <polarity> | |
| **Command Parameters** | <polarity> = NORMal \| INVert | |
| ***RST Value** | NORM | |
| **Query Syntax** | OUTPut[:TTLTrig]:POLarity? | |
| **Query Parameters** | None | |
| **Query Response** | NORM \| INV | |
| **Description** | When the voltmeter finishes a single reading or a group of readings, it can generate a completed signal on one of the VXIbus TTL trigger lines. The trigger line used is set by the OUTPut:TTLTrig command. The output is enabled by the OUTPut:STATe command. The type of completed signal (completed a single reading or completed a group of readings) is set by the OUTPut:TYPE command. The OUTPut:POLarity command sets the polarity of the voltmeter complete signal.<br><br>A polarity of NORMal says the TTLTrig line goes low on completion. A polarity of INVert says the TTLTrig line goes high on completion. | |
| **Examples** | **Command / Query** | **Response *(Description)*** |
| | `OUTP:POL INV`<br>`OUTP:POL?` | <br>INV |
| **Related Commands** | OUTPut:STATe<br>OUTPut:TYPE<br>OUTPut:TTLTrig | |

# OUTPut[:TTLTrig]:STATe

| | | |
|---|---|---|
| **Purpose** | Enables routing the voltmeter completed signal to the VXIbus TTL trigger lines | |
| **Type** | Setting | |
| **Command Syntax** | OUTPut[:TTLTrig] <n>:STATe <mode> | |
| **Command Parameters** | <n> = 0 - 7<br><mode> = OFF \| 0 \| ON \| 1 | |
| ***RST Value** | 0 | |
| **Query Syntax** | OUTPut[:TTLTrig]:STATe? | |
| **Query Parameters** | None | |
| **Query Response** | 0 \| 1 | |
| **Description** | When the voltmeter finishes a single reading or a group of readings, it can generate a completed signal on one of the VXIbus TTL trigger lines. The trigger line used is set by the OUTPut:TTLTrig command. The output is enabled by the OUTPut:STATe command. The type of completed signal (completed a single reading or completed a group of readings) is set by the OUTPut:TYPE command. The OUTPut:POLarity command sets the polarity of the voltmeter complete signal.<br><br>An OUTPut:STATe of 0 or OFF disables generation of a completion signal. An OUTPut:STATe of 1 or ON enables the generation of a completion signal. | |
| **Examples** | **Command / Query** | **Response *(Description)*** |
| | `OUTP:STAT ON`<br>`OUTP:STAT?` | <br>1 |
| **Related Commands** | OUTPut:POLarity<br>OUTPut:TYPE<br>OUTPut:TTLT | |

# OUTPut:TTLTrig:TYPE

| | | |
|---|---|---|
| **Purpose** | Determines what type of voltmeter completion signal is generated | |
| **Type** | Setting | |
| **Command Syntax** | OUTPut[:TTLTrig]:TYPE <trig_type> | |
| **Command Parameters** | <trig_type> = SINGle \| GRP \| GRoup | |
| ***RST Value** | GR | |
| **Query Syntax** | OUTPut[:TTLTrig]:TYPE? | |
| **Query Parameters** | None | |
| **Query Response** | SING \| GR | |
| **Description** | When the voltmeter finishes a single reading or a group of readings, it can generate a completed signal on one of the VXIbus TTL trigger lines. The trigger line used is set by the OUTPut:TTLTrig command. The output is enabled by the OUTPut:STATe command. The type of completed signal (completed a single reading or completed a group of readings) is set by the OUTPut:TYPE command. The OUTPut:POLarity command sets the polarity of the voltmeter complete signal.<br><br>A type of SINGle says a completion signal is sent after each voltage is measured. A type of GRoup says a completion signal is sent after all channels in the scan have been measured. | |
| **Examples** | **Command / Query** | **Response *(Description)*** |
| | `OUTP:TYPE SING`<br>`OUTP:TYPE?` | SING |
| **Related Commands** | OUTPut:STATe<br>OUTPut:TTLTrig<br>OUTPut:POLarity | |

# SAMPle:COUNt

| | | |
|---|---|---|
| **Purpose** | Sets the number of readings taken with each trigger event | |
| **Type** | Setting | |
| **Command Syntax** | SAMPle:COUNt <number> | |
| **Command Parameters** | <number> = 1 to 16383 | |
| ***RST Value** | 16 | |
| **Query Syntax** | SAMPle:COUNt? | |
| **Query Parameters** | None | |
| **Query Response** | 1 to 16383 | |
| **Description** | The Sample Count command sets the number of readings to be taken with each trigger event. For example, if the channel list has 17 channels to be read and the list should be scanned 5 times with each trigger, then the sample count should be set to 17 * 5 = 85.<br><br>NOTE: The commands MEASure and CONFigure:VOLTage automatically set the sample count to the number of channels in the channel list. | |
| **Examples** | **Command / Query** | **Response *(Description)*** |
| | `SAMP:COUN 85`<br>`SAMP:COUN?` | <br>85 |
| **Related Commands** | MEASure:VOLTage<br>CONFigure:VOLTage | |

# SAMPle:TIMer

| | | |
|---|---|---|
| **Purpose** | Sets the sample timer period | |
| **Type** | Setting | |
| **Command Syntax** | SAMPle:TIMer <period> | |
| **Command Parameters** | <period> = 1.0e-3 to 5.12e-1 | |
| ***RST Value** | 1.0e-3 | |
| **Query Syntax** | SAMPle:TIMer? | |
| **Query Parameters** | None | |
| **Query Response** | 0.001 to 0.512 | |
| **Description** | The Sample Timer command sets the period of the sample timer. The sample timer is used to pace the rate at which the scanning voltmeter scans through its channel list. | |
| **Examples** | **Command / Query** | **Response *(Description)*** |
| | `SAMP:TIM 1e-3`<br>`SAMP:TIM?` | 0.001 |
| **Related Commands** | SAMPle:SOURce | |

# TRACe[:DATA]:FEED

| | | |
|---|---|---|
| **Purpose** | Sets how data is retrieved from the voltmeter | |
| **Type** | Setting | |
| **Command Syntax** | TRACe[:DATA]:FEED <feed_type> | |
| **Command Parameters** | <feed_type> = REGister \| OFF | |
| ***RST Value** | OFF | |
| **Query Syntax** | TRACe[:DATA]:FEED? | |
| **Query Parameters** | None | |
| **Query Response** | REG \| OFF | |
| **Description** | Data (measurements) are retrieved from the voltmeter using two different methods. For most systems the easiest method to use is word serial data transfers. This is the same technique used to program the module and query the module's settings. When fast response time is important, measurements can be retrieved using register accesses. The two methods are exclusive. Only one method can be used at a time.<br><br>Setting TRACe:FEED to OFF is the way to use word serial data transfers. Setting TRACe:FEED to REGister is the way to use register accesses.<br><br>NOTE: When TRACe:FEED is set to REGister the commands FETCh? and MEASure will generate errors. This is because these are the commands that request word serial data transfers.<br><br>See the section on register access for more information. | |
| **Examples** | **Command / Query** | **Response *(Description)*** |
| | `TRAC:FEED REG`<br>`TRAC:FEED?` | REG |
| **Related Commands** | None | |

# TRIGger[:IMMediate]

| | | |
|---|---|---|
| **Purpose** | Causes a trigger event to occur for the scanning voltmeter | |
| **Type** | Event | |
| **Command Syntax** | TRIGger[:IMMediate] | |
| **Command Parameters** | None | |
| ***RST Value** | None | |
| **Query Syntax** | N/A | |
| **Query Parameters** | N/A | |
| **Query Response** | N/A | |
| **Description** | The Trigger command will cause the scanning voltmeter to start taking readings when in the waiting-for-trigger mode as determined by the scan list and the sample count commands. | |
| **Examples** | **Command / Query** | **Response *(Description)*** |
| | `TRIG` | |
| **Related Commands** | J<br>IT | |

# TRIGger[:SEQuence]:SLOPe

| | | |
|---|---|---|
| **Purpose** | Sets the active edge of the voltmeter trigger | |
| **Type** | Setting | |
| **Command Syntax** | TRIGger[:SEQuence]:SLOPe <slope> | |
| **Command Parameters** | <slope> = POSitive \| NEGative | |
| ***RST Value** | NEGative | |
| **Query Syntax** | TRIGger[:SEQuence]:SLOPe? | |
| **Query Parameters** | None | |
| **Query Response** | POS \| NEG | |
| **Description** | The voltmeter must be INITiated and triggered to make measurements. The TRIGger:SOUR command selects the source of the trigger. The TRIGger:SLOPe command selects the active edge of the trigger<br><br>If POSitive is selected then the rising edge of the trigger signal triggers the voltmeter. If NEGative is selected then the falling edge of the trigger signal triggers the voltmeter. | |
| **Examples** | **Command / Query** | **Response *(Description)*** |
| | `TRIG:SLOP POS`<br>`TRIG:SLOP?` | <br>POS |
| **Related Commands** | INItiate<br>TRIGger:SOURce<br>TRIGger | |

# TRIGger[:SEQuence]:SOURce

| | | |
|---|---|---|
| **Purpose** | Sets the source of the voltmeter trigger | |
| **Type** | Setting | |
| **Command Syntax** | TRIGger[:SEQuence]:SOURce <source> | |
| **Command Parameters** | <source> = IMMediate \| EXTernal \| TTLTrig<n><br><n> = 0 \| 1 \| 2 \| 3 \| 4 \| 5 \| 6 \| 7 | |
| ***RST Value** | IMM | |
| **Query Syntax** | TRIGger[:SEQuence]:SOURce? | |
| **Query Parameters** | None | |
| **Query Response** | IMM \| EXT \| 0 \| 1 \| 2 \| 3 \| 4 \| 5 \| 6 \| 7 | |
| **Description** | The voltmeter must be INITiated and triggered to make measurements. The TRIGger:SOUR command selects the source of the trigger. The TRIGger:SLOPe command selects the active edge of the trigger<br><br>If IMMediate is selected then the trigger must come from a TRIGger command. If EXTernal is selected then the trigger comes from the front panel connector identified as the TRIGGER INPUT. | |
| **Examples** | **Command / Query** | **Response *(Description)*** |
| | `TRIG:SOUR EXT`<br>`TRIG:SOUR?` | <br>EXT |
| **Related Commands** | TRIGger<br>TRIGger:SLOPe<br>INITiate | |

# TRIGger:TYPE

| | | |
|---|---|---|
| **Purpose** | Sets how the module operates after a trigger | |
| **Type** | Setting | |
| **Command Syntax** | TRIGger:TYPE <trig_type> | |
| **Command Parameters** | <trig_type> = ONCE \| MULTiple | |
| ***RST Value** | ONCE | |
| **Query Syntax** | TRIGger:TYPE? | |
| **Query Parameters** | None | |
| **Query Response** | ONCE \| MULT | |
| **Description** | Refer to the data collection sequence diagram in the examples section. This command determines what to do after SCAN_COUNT data is collected. If ONCE is set, the module changes to the IDLE state. If MULTiple is set, the module changes back to the Waiting for Trigger state. | |
| **Examples** | **Command / Query** | **Response *(Description)*** |
| | `TRIG:TYP MULT`<br>`TRIG:TYPE?` | MULT |
| **Related Commands** | INITiate<br>TRIGger<br>SAMPle:COUNt | |

# REQUIRED SCPI COMMANDS

## STATus:OPERation:CONDition?

| | | |
|---|---|---|
| **Purpose** | Queries the Operation Status Condition Register | |
| **Type** | Required SCPI command | |
| **Command Syntax** | None – Query Only | |
| **Command Parameters** | N/A | |
| ***RST Value** | N/A | |
| **Query Syntax** | STATus:OPERation:CONDition? | |
| **Query Parameters** | None | |
| **Query Response** | 0 | |
| **Description** | The Operation Status Condition Register query is provided for SCPI compliance only. The VM2716A does not alter the state of any of the bits in this register and always reports a 0. | |
| **Examples** | **Command / Query** | **Response *(Description)*** |
| | `STAT:OPER:COND?` | 0 |
| **Related Commands** | None | |

# STATus:OPERation:ENABle

| | | |
|---|---|---|
| **Purpose** | Sets the Operation Status Enable Register | |
| **Type** | Required SCPI command | |
| **Command Syntax** | STATus:OPERation:ENABle <NRf> | |
| **Command Parameters** | <NRf> = numeric ASCII value from 0 to 32767 | |
| ***RST Value** | NRf must be specified | |
| **Query Syntax** | STATus:OPERation:ENABle? | |
| **Query Parameters** | None | |
| **Query Response** | Numeric ASCII value from 0 to 32767 | |
| **Description** | The Operation Status Enable Register is included for SCPI compatibility and the VM2716A does not alter any of the bits in this register. The register layout is as follows:<br><br>Bit 0 - Calibrating<br>Bit 1 - Setting<br>Bit 2 - Ranging<br>Bit 3 - Sweeping<br>Bit 4 - Measuring<br>Bit 5 - Waiting for trigger<br>Bit 6 - Waiting for arm<br>Bit 7 - Correcting | |
| **Examples** | **Command / Query** | **Response *(Description)*** |
| | `STAT:OPER:ENAB 0` | 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 VM2716A does not alter any of the bits in this register and always reports a 0. | |
| **Examples** | **Command / Query** | **Response *(Description)*** |
| | STAT:OPER? | |
| **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 VM2716A 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?` | |
| **Related Commands** | None. | |

# STATus:QUEStionable:ENABle?

| | | |
|---|---|---|
| **Purpose** | Sets the Questionable Status Enable Register | |
| **Type** | Required SCPI command | |
| **Command Syntax** | STATus:QUEStionable:ENABle <NRf> | |
| **Command Parameters** | <NRf> = numeric ASCII value from 0 to 32767 | |
| ***RST Value** | NRf must be supplied | |
| **Query Syntax** | STATus:QUEStionable:ENABle? | |
| **Query Parameters** | None | |
| **Query Response** | Numeric ASCII value from 0 to 32767 | |
| **Description** | The Status Questionable Enable command sets the bits in the Questionable Status Enable Register. This command is provided only to comply with the SCPI standard.<br><br>The Status Questionable Enable query reports the contents of the Questionable Status Enable Register. The VM2716A 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`<br>`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 VM2716A does not alter the bits in this register and queries always report a 0. | |
| **Examples** | **Command / Query** | **Response *(Description)*** |
| | `STAT:QUES?` | |
| **Related Commands** | None | |

# SYSTem:ERRor?

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

# SYSTem:VERSion?

| | | |
|---|---|---|
| **Purpose** | Queries the SCPI version number the VM2716Acomplies with | |
| **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 version of the SCPI standard with which the VM2716A complies. | |
| **Examples** | **Command / Query** | **Response *(Description)*** |
| | SYST:VERS? | 1994.0 |
| **Related Commands** | None | |

# SECTION 5

# THEORY OF OPERATION

## INTRODUCTION

The VM2716A provides the functionality of a 4.5 digit DC voltmeter and a FET multiplexer. The VM2716A can be programmed to scan through its channels list upon receipt of a word serial trigger or by programming the internal sample timer from 1ms to 512 ms, allowing for up to 1000 readings per second at 4.5 digits of resolution. Aperture times may also be programmed from 400 $\mu$s to 20 ms allowing for the rejection of 50 Hz or 60 Hz noise.

To synchronize to other instruments within the VXIbus chassis the VM2716A can also be programmed to generate a sync pulse to any of the TTL trigger lines upon completion of operation.

The VM2716A is composed of four sections: the input and front end, calibration and measurement, timing and control, and power forms generation and regulation. The following sections of this chapter will discuss the theory of operation for each of the sections.

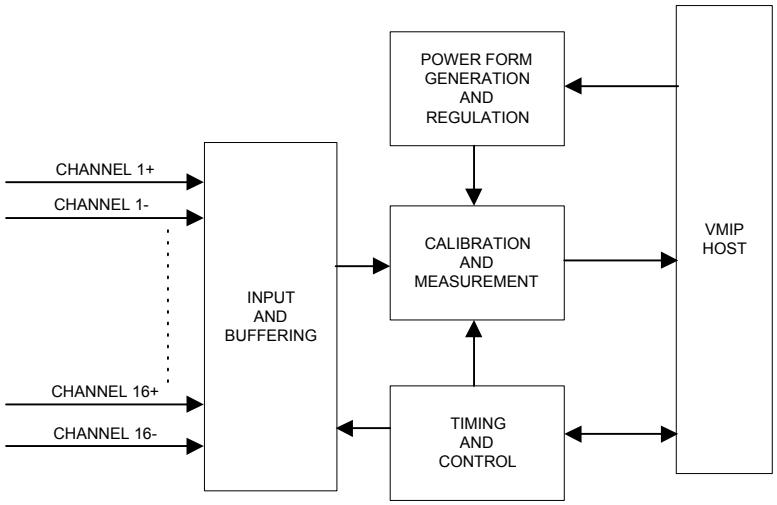



**FIGURE 5-1 VM2716A BLOCK DIAGRAM**

## INPUT AND BUFFERING CIRCUITRY

The input resistor network provides a line-to-line input impedance of 10 MΩ (See Figure 5-2). The FET switches have internal diode protection and can withstand input over voltages up to ±250.0 VDC or peak AC between differential pairs or ±125.0 VDC or peak AC from any input to ground. The FET switches (U2 and U3) provide for the switching of inputs to the front end (U7a and U7b) of the measurement circuitry. Analog switches U8 and U11 provide the selection of feedback resistors for amplifier gains of 1.0, 10.0 and 100.0. An amplifier gain of 1.0 corresponds to the input voltage range of ±20.0 V, a gain of 10 is for ±2.0 V range and a gain of 100.0 is for the ±200 mV range. Z1 is a precision resistor network used to provide the feedback for U7a and U7b. Z1 provides for a gain of 1.0 or 10.0. The output of U7 is routed to the instrumentation amplifier U9. The offset for U9 is set up by Z2, and R112. R112 is factory-adjusted and set and does not require further adjusting. If Z2 requires replacement, then R112 will require readjusting. The output of U9 is routed to the instrumentation amplifier U10. The analog switch U11 selects a gain of 1.0 or a gain of 10.0. Z3 is a precision resistor network providing the feedback for U10. R113 is factory set to zero the output of U10 through adjusting the offset at the inverting input. Unless Z3 has been replaced, this potentiometer R113 requires no adjusting. Filter capacitors are used in all feedback circuits to reduce possible oscillations during high-speed switching.

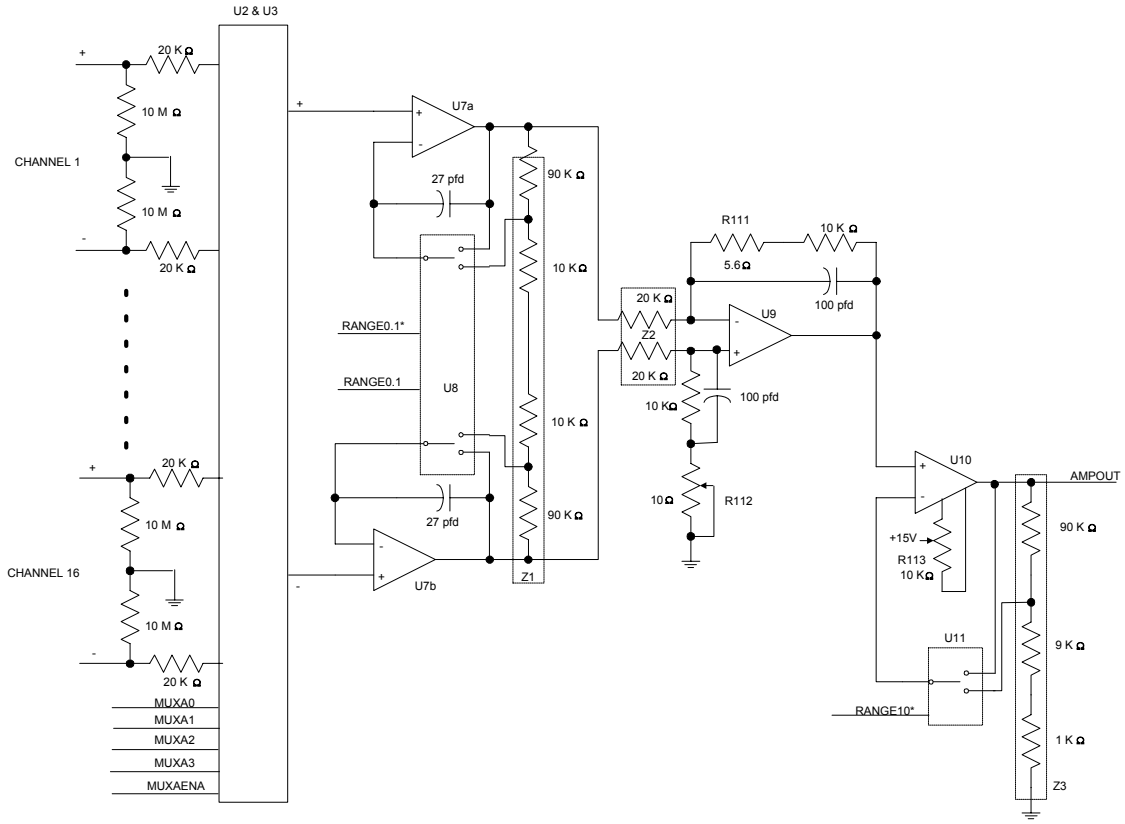



**FIGURE 5-2. INPUT AND BUFFERING CIRCUITRY**

For an example of how this circuitry works, it will be assumed that a single-ended signal that has a voltage level of 3.5 V using Channel 1 as the input is being measured. The input on Channel 1 high side is 3.5 VDC and the low side is externally tied to ground. The range is set for ±20.0 VDC. The select lines MUXA0 through MUXA3 and MUXBENA are all set low and MUXAENA is set high. This enables MUXA (U2) to pass Channel 1 to the instrumentation amplifier U7a and U7b. U8 control signals RANGE0.1* and RANGE0.1 are set to a low. This configures the feedback circuitry for U7a and U7b for a gain of 1. The output of U7a is 3.5 VDC and the output of U7b is 0.0 V, ground potential. The outputs of U7a and U7b are sent to the differential amplifier U9. U9 has a gain of 1.0 and buffers the difference between U7a and U7b. The output of this differential amplifier is then sent to instrumentation amplifier U10. U10's gain is set by analog switch U11. U11's control signals RANGE10 and RANGE10* are set high, thus configuring U10 for a gain of 1.0. The output of U10 is now normalized at 3.5 VDC. The normalized signal is now sent to the calibration and measurement section.

## CALIBRATION AND MEASUREMENT CIRCUITRY

The VM2716A utilizes an octal eight-bit trim DAC for the calibration of the input signal ADC's gain and offset values. Two outputs of the calibration DAC are used for gain adjustments and two for offset adjustments. Of the two, one is for fine adjustments and one is for coarse adjustments. The fine adjustment contains a 100 kΩ resistor in series with the DAC output and the instrumentation amplifier's input, where the coarse does not. The calibration DAC, U15, requires a precision 5.0 V reference that is supplied by U17 (See Figure 5-3). U17 has an output voltage drift over the temperature range of 0°C to 70°C of 2 parts per million (ppm). Since the calibration DAC and reference buffer, U16a, requires more current than the precision reference can reliably supply, a current amplifier section composed of U16d and Q8 is used. The 5.0 V precision reference is also used to set up the input reference voltage of 2.5 V to U14 through the use of a precision laser trim voltage divider network of Z4.

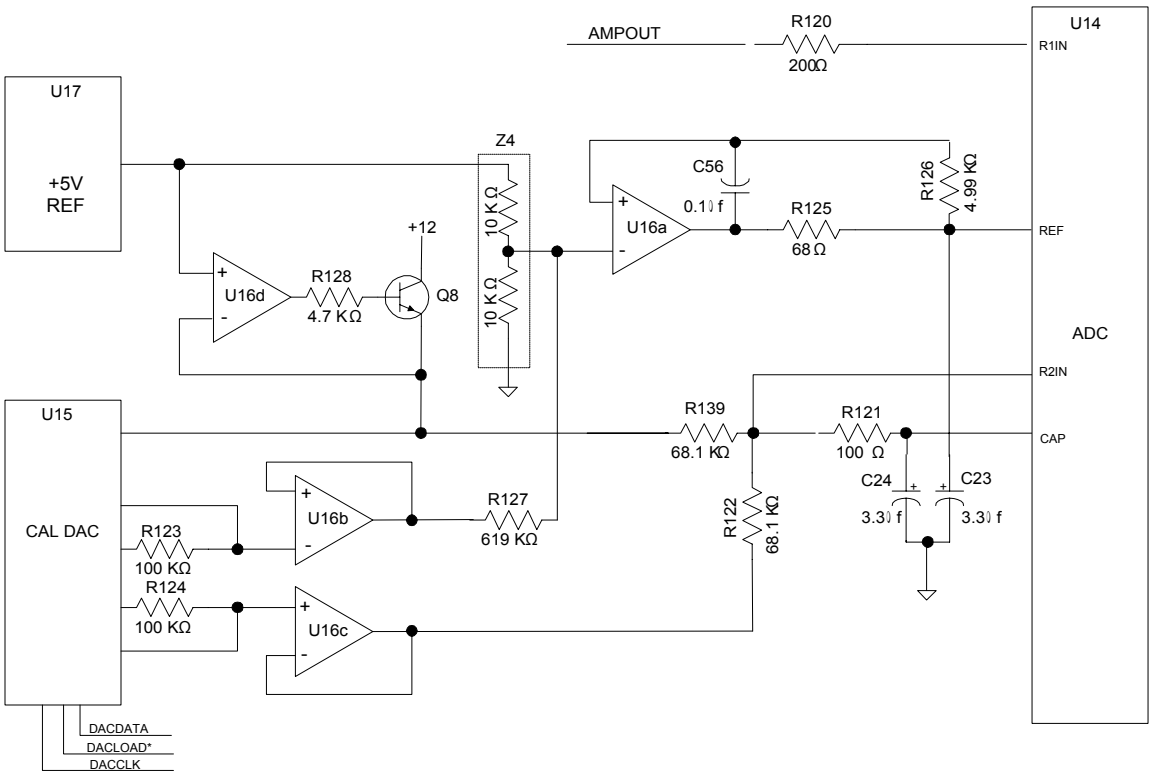



**FIGURE 5-3 CALIBRATION AND MEASUREMENT CIRCUITRY**

The gain adjust output of the calibration DAC is buffered by U16b and is used to slightly adjust the input reference voltage. U16a then buffers the input reference voltage to U14. C56 on the output of U16a is used to bypass high-frequency switching oscillations induced by C23 on the REF input of U14. The input voltage reference range is 2.3 V to 2.7 V and determines the actual size of the ADC LSB. Increasing the reference voltage will increase the full-scale range and the LSB size of the converter that can improve the Signal to Noise Ratio (SNR). The calculation for determining the output voltage of the calibration DAC is determined as follows:

$$V_{OUT} = D \times (V_{REF}H - V_{REF}L)/256 + V_{REF}L$$

Where D is the digital word value, $V_{REF}H$ is +5.0 V and $V_{REF}L$ is ground or 0.0 V.

An example of what the output voltage would be if D = 16 is:

$$V_{OUT} = 16 \times (5 - 0)/256 + 0$$
$$V_{OUT} = 16 \times 0.01953$$
$$V_{OUT} = 0.31248 \text{ V}$$

If $V_{OUT}$ is known but the digital word is not, the equation can be rearranged to read:

$$D = V_{OUT}/((V_{REF}H - V_{REF}L)/256-V_{REF}L)$$

An example of what D would be if the output voltage = 0.70308 is:

$$D = V_{OUT} /((V_{REF}H - V_{REF}L)/256-V_{REF}L)$$
$$D = 0.70308/ ((5 - 0)/256 - 0)$$
$$D = 0.70308/(5/256)$$
$$D = 0.70308/ 0.01953$$
$$D = 36$$

The CAP pin on U14 is the output of the ADC internal reference buffer. This signal is adjusted to a level of 2.5 V and 5.0 V by the use of the offset amplifier U16c and the divider network of R139 and R122. The previous calculations apply to this circuit as well. The calibration DAC is serially loaded by the timing and control FPGA and are covered in the next section.

## TIMING AND CONTROL FPGA

The timing and control FPGA has all the logic necessary to interface with the VMIP host and the VM2716A circuitry. The FPGA contains the aperture counter, conversion cycle counter and conversion logic, data bus interface, clock generation and VXI bus trigger logic (See Figure 5-4).

The clock generation logic divides the VXIbus 10 MHz clock by 250. The resultant 40 kHz clock is used to gate the aperture counter and conversion logic. The aperture counter is a 16-bit user programmable counter used to tell the control logic how long to sample each measurement. The aperture counter is used for the rejection of noise components on the measured signal. An aperture counter value of 16.7 ms is used to reject 60 Hz noise and a value of 20 ms is for the rejection of 50 Hz noise. The conversion counter and conversion logic circuitry are used to allow for the proper settling time of the ADC (U14) to elapse before latching the converted data into the FIFO. When the conversion counter has reached its top count, it issues a signal to the conversion logic. This signal is used by the conversion logic to toggle the normally high DACBYTE signal to the ADC. The DACBYTE signal is used by the ADC to output the converted data to the FIFO. When the DACBYTE is high, the ADC will output the most significant byte to the FIFO, and when low, the least significant byte. The conversion logic also generates the FIFOWRITE signal to the FIFO in order to latch the data. When this signal goes low, it causes the FIFO to latch the converted data byte and increment its internal address counter to the next address.

The timing and control FPGA does not need to generate interrupts to the VMIP host because of the relative short conversion time required by the ADC. The timing and control FPGA's conversion circuitry generates a BUSY signal instead to the VMIP host. The VMIP host waits on the BUSY signal to go low for the determination of the completion of the conversion cycle. This conversion cycle time is usually in the 250 $\mu$s range but varies on the aperture time specified by the user.

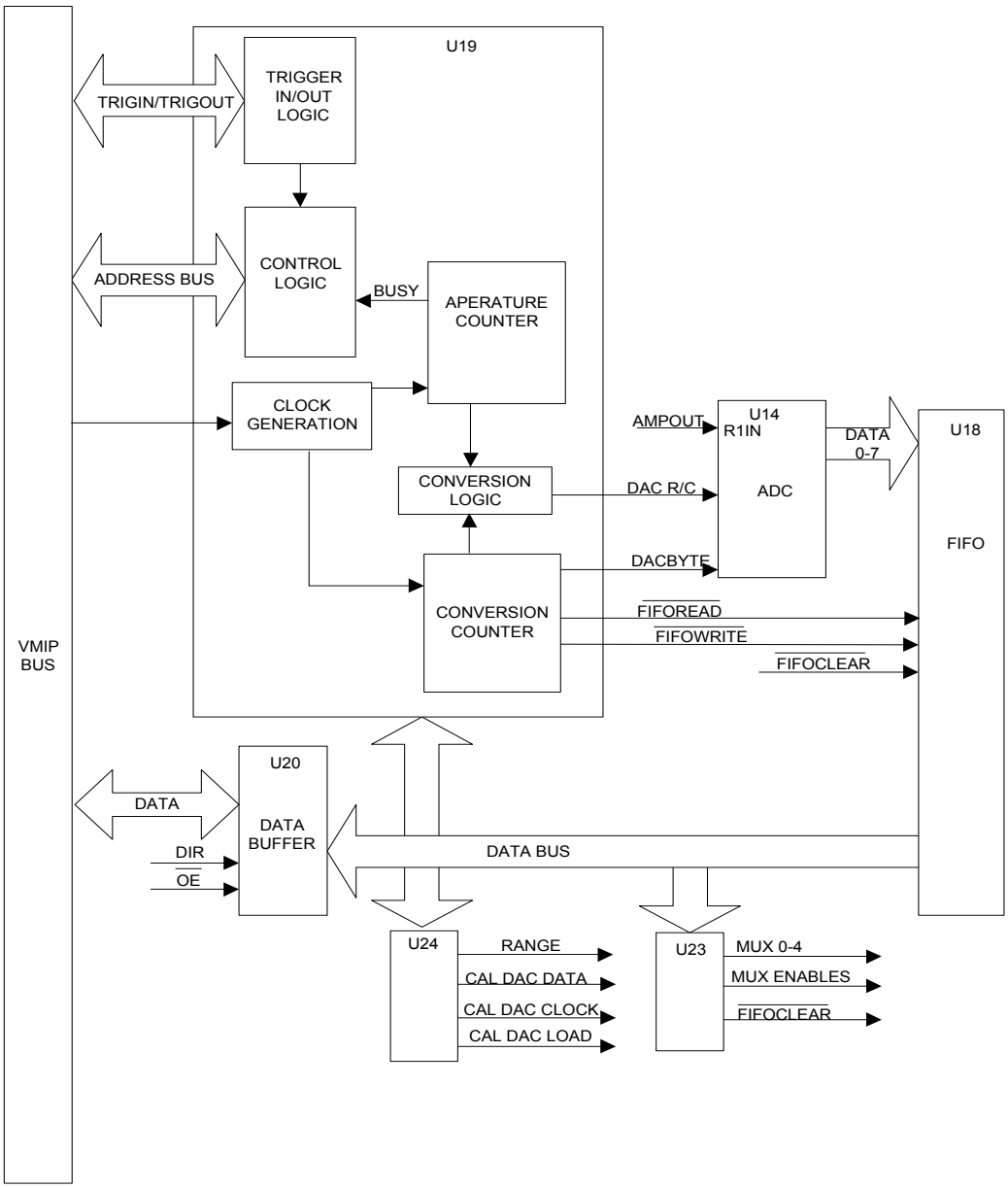



**FIGURE 5-4 TIMING AND CONTROL FUNCTIONAL BLOCK DIAGRAM**

When a VXIbus trigger or a word serial event occurs DACR/C is pulled low. The clock generation circuitry is loaded and initiated. The aperture counter starts to count up. This pulls BUSY high notifying the VMIP host that a conversion cycle has commenced. When the aperture counter has reached its terminal count, an END signal is issued to the conversion logic. This signal will cause the DACR/C signal to go low causing the ADC to convert and lowering the BUSY signal and starting the conversion counter. When the conversion counter has reached its terminal count, the conversion logic will pull the FIFOWRITE signal low. This will cause the most significant byte of the converted data to be latched into the FIFO. The FIFO's internal address counter will increment to the next address and wait for the next FIFOWRITE signal. The conversion logic will then raise the DACBYTE signal causing the ADC to output the least significant bit to the FIFO. The conversion logic then lowers the FIFOWRITE signal latching the data. The timing and control FPGA is now back in its initial state prior to receiving a trigger or word serial command. This sequence will continue until all measurements in the task list have been completed.

Data is read from the FIFO one byte at a time and is under the control of the timing and control FPGA. When a read is initiated by the VMIP host the FPGA decodes the address and data bits from the VMIP host and will generate the output enable signal FIFORD0*. This signal causes the FIFO to output the first most significant byte that was loaded onto the data bus. The FIFO's internal address counter then decrements and the VMIP host retrieves the data. This sequence continues until all the captured data has been retrieved or the FIFOEMPTY has been reached.

In addition to the previous functions, the FPGA also contains the logic necessary for the selection of which FET switch is to be used and calibration DAC control.

The calibration DAC functions control the loading of the calibration DAC. The calibration DAC, as previously mentioned, is an octal 8-bit DAC and four of the eight DACs are used. The FPGA receives the data and which DAC is to be used from the VMIP host. The FPGA outputs a clock and a serial data stream (MSB first) and pulls down the DACLOAD* line. The calibration DAC is loaded one DAC at a time. The first three bits of the serial data contain the internal DAC's address and the remaining eight bits are data.

## POWER FORMS GENERATION AND REGULATION

The instrumentation amplifier U7 and the FET switches U2 and U3 require a stable ±20 VDC source for proper operation. This is achieved by the differential amplifier, U6a, and transistor, Q7, for the +20.0 VDC side and differential amplifier U6b and transistor Q6 for the -20 VDC side (See Figure 5-5). Since both the positive and negative circuitry operate identically, only the positive side will be observed. The VXI backplane ±24 VDC is divided by R109, R107 and R101. The non-inverting input of U6a will be 5.9 VDC. This is set up by the divider network of R107, R101 and R103. If the non-inverting input of U6a starts to rise above 5.9 VDC, the output of U6a will start to go more positive. This positive voltage will cause Q7 to conduct and pull down the voltage at the junction between R101 and R107. This in turn lowers the non-inverting input voltage thereby keeping the voltage at the top of R101 at +20 VDC. Zener diodes D1, D2, D3 and D4 are added for extra protection and capacitors are used for filtering the ±20 VDC.

For reliable, noise-free operation, the ADC (U14) requires its own +5 VDC power source. Q5 is used to generate the +5 VDC for U14 from the VXI backplane +12 VDC. Capacitor C11 is used for better filtering of the input +12 VDC.

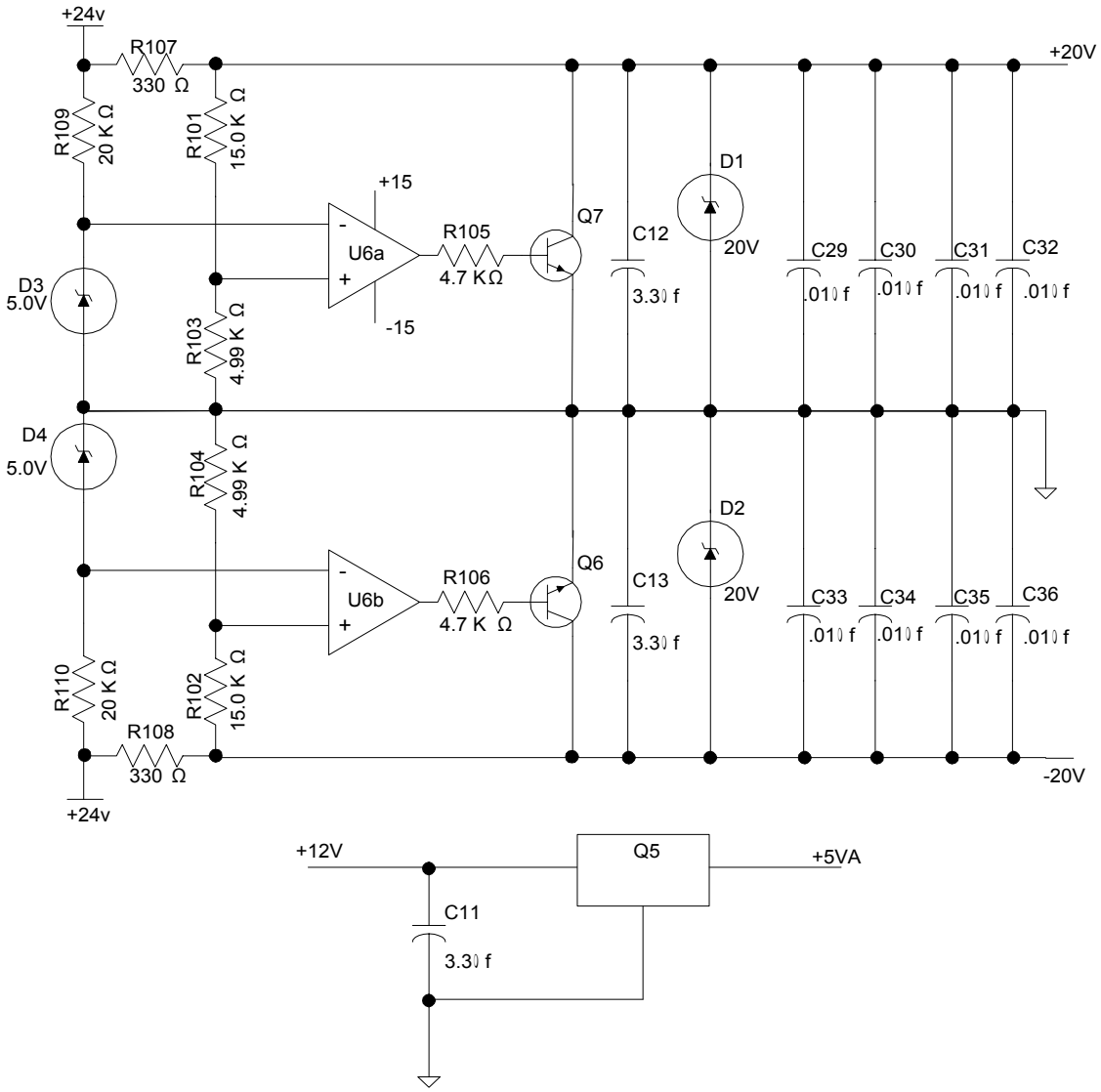



**FIGURE 5-5 POWER FORMS GENERATION AND REGULATION**

# INDEX


*

*CLS....60
*ESE....61
*ESR?....62
*IDN?....63
*OPC....62, 64, 69
*RST....65
*STB?....66
*TRG....67
*TST?....68
*WAI....69

*A*

abort....35
ABORt....70
ADC....113
address counter....113
aperture....109, 113
aperture counter....113, 115
aperture time....79

*B*

backplane....115
backplane jumpers....17, 18
based word serial interface....13
BUSY signal....113, 115

*C*

calibration 14, 26, 71, 72, 73, 74, 77, 78, 109, 111, 112, 113
calibration DAC....115
Calibration Gain....46
Calibration Offset....46
calibration security....26, 73, 75, 78
CALibration:COUNt....71
CALibration:COUNt?....22
CALibration:DEFault....23, 72
CALibration:Gain....73
CALibration:RESet....25, 74
CALibration:SECure:CODE....75
CALibration:SECure:CODE<code>....26
CALibration:SECure:STATe....76
CALibration:SECure:STATe,mode[,<code>]....27
CALibration:STORe....28, 77
CALibration:ZERO....78
CALibration<adjust>:GAIN<value>....24
CALibration<adjust>:ZERO <value>....29
channel list....87
clock generation....113, 115
CLS....56
command strings....46
Configure Voltage....35
CONFigure:[VOLTage:]APERture....79
CONFigure:[VOLTage:]APERture<time>....30
CONFigure:VOLTage....80
CONFigure:VOLTage<range>,<channel_list>....31
connector....19, 20
control....109, 113, 115
conversion counter....115
conversion cycle....115
conversion cycle counter....113
conversion logic....113, 115
cooling....17

*D*

data bus interface....113
data collection....98
direct register access....13
dynamic address configuration....18

*E*

ESE....56
ESR....56

*F*

feedback circuits....110
Fetch....35, 87
FETCh?....32, 81, 82
FIFO....113, 115
filtering....116
FLASH memory....81
format....83
FORMat:VOLTage:DATA....83
FORMat:VOLTage[:DATA]<length>....33
FPGA....113, 115

*G*

gain....14, 23, 73, 110, 111, 112

*I*

IDN?....56
IEEE 488.2 commands....55
IEEE 488.2 Common Commands....56
Initiate....32, 35, 87
INITiate....34, 84
INITiate and TRIGger....85
input impedance....110
input reference voltage....112
input resistor....110
input signal....111
Instrument Specific SCPI Commands....57
interface....19

*L*

logic....113
logical address....17, 18
Logical Address....14

*M*

MEASure:VOLTage....87
MEASure:VOLTage<range>,<channel_list>....35
measurement....109, 111
measurement circuitry....110
message-based....13, 14

### *O*

offset ..... 14, 23, 29, 78, 111
OPC ..... 56
output ..... 110
output buffer ..... 82
OUTPut:TTLTrg ..... 90
OUTput:TTLTrig ..... 88
OUTPut:TTLTrig <line> ..... 36
OUTPut:TTLTrig:POLarity ..... 89
OUTPut:TTLTrig:POLarity <polarity> ..... 37
OUTPut:TTLTrig:TYPE ..... 91
OUTPut:TTLTrig<n>[:STATe]<mode> ..... 38
OUTPut:TYPE <trig type> ..... 39

### *P*

polarity ..... 89
power ..... 17, 18
programmable counter ..... 113

### *R*

register access ..... 48, 94
Required SCPI commands ..... 55
RST ..... 55, 56, 57, 58

### *S*

sample ..... 92, 93, 113
SAMPle:COUNt ..... 92
SAMPle:COUNT<number> ..... 40
SAMPle:TIMer ..... 93
SAMPle:TIMer<period> ..... 41
scan list ..... 95
scanning ..... 41, 43, 80, 87
SCPI Required Commands ..... 58
security ..... 76
security code ..... 27, 76
security state ..... 27
serial data ..... 115
STATus:OPERation:CONDition? ..... 99
STATus:OPERation:ENABle ..... 100
STATus:OPERation[:EVENt]? ..... 101
STATus:PRESet ..... 102
STATus:QUEStionable:CONDition? ..... 103
STATus:QUEStionable:ENABle ..... 104
STATus:QUEStionable:EVENt ..... 105
STB? ..... 56
sync pulse ..... 109
SYSTem:ERRor? ..... 106
SYSTem:VERSion? ..... 107

### *T*

temperature range ..... 111
terminal count ..... 115
timing ..... 109, 113, 115
timing and control FPGA ..... 113, 115
trace data ..... 42
TRACe:DATA:FEED <feed type> ..... 42
TRACe[:DATA]:FEED ..... 94
TRG ..... 56
trigger ..... 35, 43, 84, 88, 89, 90, 91, 95, 96, 97, 98, 109, 115
TRIGger ..... 86
trigger logic ..... 113
TRIGger:[IMMediate] ..... 95
TRIGger:SEQuence:SLOPe <slope> ..... 44
TRIGger:SEQuence:SOURce <source> ..... 45
TRIGger[:IMMediate] ..... 43
TRIGger[:SEQuence]:SLOPe ..... 96
TRIGger[:SEQuence]:SOURce ..... 97
TST? ..... 56
TTL trigger ..... 36, 37, 38, 39

### *V*

VMIP ..... 13, 14, 18
voltage range ..... 80, 87, 110
VXIbus ..... 13, 14, 17

### *W*

WAI ..... 56
word serial data transfer ..... 94
word serial event ..... 115
word serial trigger ..... 109

**Artisan Technology Group** is an independent supplier of quality pre-owned equipment

## Gold-standard solutions

Extend the life of your critical industrial, commercial, and military systems with our superior service and support.

## We buy equipment

Planning to upgrade your current equipment? Have surplus equipment taking up shelf space? We'll give it a new home.

## Learn more!

Visit us at **artisantg.com** for more info on price quotes, drivers, technical specifications, manuals, and documentation.

Artisan Scientific Corporation dba Artisan Technology Group is not an affiliate, representative, or authorized distributor for any manufacturer listed herein.

We're here to make your life easier. How can we help you today?

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

ARTISAN® TECHNOLOGY GROUP