Model 38xx
Digital Multi-Meter
90401280

All technical data and specifications in this publication are subject to change without prior notice and do not represent a commitment on the part of Giga-tronics, Incorporated.

Warranty

Giga-tronics Series 3000 Switching Modules are warranted against defective materials and workmanship for three years from date of shipment, or as detailed in the warranty section of this manual. Giga-tronics will, at its option, repair or replace products that are proven defective during the warranty period. This warranty DOES NOT cover damage resulting from improper use, nor workmanship other than Giga-tronics service. There is no implied warranty of fitness for a particular purpose, nor is Giga-tronics liable for any consequential damages. Specification and price change privileges are reserved by Giga-tronics.

CONTACT INFORMATION

Giga-tronics, Incorporated
4650 Norris Canyon Road
San Ramon, California 94583
Telephone: 800.726.4442 (only within the United States)
925.328.4650
Fax: 925.328.4700
On the Internet: www.gigatronics.com

Regulatory compliance information

This product complies with the essential requirements of the following applicable European Directives, and carries the CE mark accordingly.

89/336/EEC and 73/23/EEC	EMC Directive and Low Voltage Directive
EN61010-1 (1993)	Electrical Safety
EN61326-1 (1997)	EMC – Emissions and Immunity
Manufacturer's Name:	**Manufacturer's Address**
Giga-tronics, Incorporated	4650 Norris Canyon Road San Ramon, California 94583 U.S.A.
Type of Equipment:	**Model Series Number**
Switching Module	38xx-xxxx

Declaration of Conformity on file. Contact Giga-tronics at the following;

Giga-tronics, Incorporated
4650 Norris Canyon Road
San Ramon, California 94583
Telephone: 800.726.4442 (only within the United States)
925.328.4650
Fax: 925.328.4700

Record of Changes to This Manual

Use the table below to maintain a permanent record of changes to this document. Corrected replacement pages are issued as Technical Publication Change Instructions (TPCI). When you are issued a TPCI, do the following:

1. Insert the TPCI at the front of the manual binder.
2. Remove the pages from the manual binder that are noted in the TPCI.
3. Replace the page(s) removed in the previous step with the corrected page(s).
4. Record the changes in the table below.

TPCI Number	TPCI Issue Date	Date Entered	Comments

Revision History			
Revision	**Description of Change**	**Chg Order #**	**Approved By**
A	Initial Release 7/03		
B	Updated 8/03		
C	Updated		
D	Updated 5/04		
E	Updated 9/09		
F	Updated 9/10		
G	Reformatted 3/12		RCW

Contents

Chapter 1 **Introduction**

1.1 Safety and Manual Conventions

This manual contains conventions regarding safety and equipment usage as described below.

1.1.1 Product Reference

Throughout this manual, the term “Common Core Switching Platform, Series 8800” refers to all models of within the series, unless otherwise specified.

1.1.2 Personal Safety Alert

WARNING: Indicates a hazardous situation which, if not avoided, could result in death or serious injury.

1.1.3 Equipment Safety Alert

CAUTION: Indicates a situation which can damage or adversely affect the product or associated equipment.

1.1.4 Notes

Notes are denoted and used as follows:

NOTE: Highlights or amplifies an essential operating or maintenance procedure, practice, condition or statement.

1.1.5 Electrical Safety Precautions

Any servicing instructions are for use by service-trained personnel only. To avoid personal injury, do not perform any service unless you are qualified to do so.

For continued protections against fire hazard, replace the AC line fuse only with a fuse of the same current rating and type. Do not use repaired fuses or short circuited fuse holders.

Chapter 2 **Configuration Table**

The 38XX instrumentation VXI modules come in a number of options.

The 3801 is a C size VXI Module which has a DMM and the ability to mount one industrial standard M-Module mezzanine unit. Specifications for the M-Modules are found in other ASCOR manuals.

- A 3801-1004 is a 3801 with a MODEL 1004 DMM.
- A 3801-1005 is a 3801 with a MODEL 1005 DMM/LCR

THE 3802 is a C size VXI Module which has a DMM but NO ability to mount M-Modules.

- A 3802-1004 is a 3802 with a MODEL 1004 DMM.
- A 3802-1005 is a 3801 with a MODEL 1005 DMM/LCR

Chapter 3 **Functional Description**

3.1 Introduction

The purpose of this manual is to provide the technical specifications for the Digital MultiMeters (DMM) use in the ASCOR 38XX series of VXI modules and other ASCOR instrumentation units.

3.2 General Description

There are two different DMM's offered with the 38XX Series VXI instrumentation modules.

The **MODEL 1004** provides standard DMM functions such as AC & DC Voltage and Current measurements and 2 and 4 wire Resistance measurements.

The **MODEL 1005** provides all the capabilities of the MODEL 1004 and in addition provides additional features such as 6 wire resistance measurements, Inductance and Capacitance measurements and AC and DC voltage and current Sourcing.

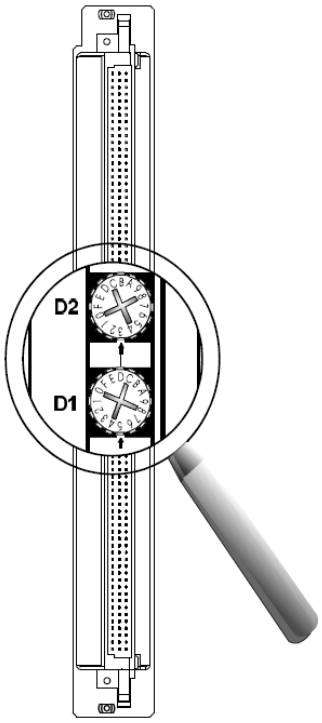
A complete summary of the features of the MODEL 1004 and MODEL 1005 are shown in the following pages. Detailed specifications are shown in the body of the manual.

Chapter 4 Controls and Indicators

The following controls and indicators are provided to select and display the functions of the ASCOR 38xx Module's operating environment.

4.1 VXI Logical Address

The Logical Address Switch is dual circular switches, D1 and D2 which are located at the rear of the module. The address can be set to any value between 1 and 255 (decimal) or 1 and FF (hexadecimal), (address 0 is reserved for the resource manager). However, the Module fully supports Dynamic Configuration as defined in ***Section F of the VXI specification,*** address 255 (FF) should be selected only if the Resource Manager also supports Dynamic Configuration.



4.2 LEDs

The following LEDs are visible at the Module's front panel to indicate the

status of the module's operation:

4.2.1 "BUS" LED

This green color LED is normally off and will flash on when the module is addressed by the system.

4.2.2 "PWR" LED

This red color LED is normally on when the Module is Powered up.

Chapter 5 Internal Settings

The following items are inside the module and can be reached by removing the side cover.

5.1 Fuse

The ASCOR VXI 38xx uses a 10 Amp fuse in the +5 Volt line and is located on the Mother Board (MB) assembly.

5.2 VXI_{bus} Interrupt Level Selection

The VXIbus interrupt level is set with three bits in the “3Eh” register.

See the section on “A16 ADDRESS SPACE REGISTER DESCRIPTION”.

The interrupt level is factory set to “no interrupt”.

Chapter 6 **Specifications**

MODEL 3801-1004 or MODEL 3802-1004 (more detailed specifications are found in Chapter 11)

DC VOLTS

- 4 ranges (330mV, 3.3V, 33V, 330V)
- Accuracy is better than 0.004%

DC CURRENT

– 4 ranges (3.3mA, 33mA, 330mA, 2.5A)
- Accuracy is better than 0.1%

AC VOLTS (RMS)

– 4 ranges (330mV, 3.3V, 33V, 250V)
- Accuracy is better than 0.9% (20 Hz to 50 KHz)

AC CURRENT (RMS)

– 4 ranges (3.3mA, 33mA, 330mA, 2.5A)
- Accuracy is better than 0.9% (20 Hz to 50 KHz)

RESISTANCE (TWO WIRE)

- 5 ranges (330Ω, 3.3 KΩ, 33 KΩ, 330 KΩ, 3.3MΩ)
- Accuracy is better than 0.003%

RESISTANCE (FOUR WIRE)

- 4 ranges (330Ω, 3.3 KΩ, 33 KΩ, 330 KΩ)
- Accuracy is better than 0.003%

MODEL 3801-1005 or MODEL 3802-1005

DC VOLTS

- 4 ranges (330mV, 3.3V, 33V, 330V)
- Accuracy is better than 0.004%

DC CURRENT

– 4 ranges (3.3mA, 33mA, 330mA, 2.5A)
- Accuracy is better than 0.1%

AC VOLTS (RMS)

– 4 ranges (330mV, 3.3V, 33V, 250V)
- Accuracy is better than 0.9% (20 Hz to 50KHz)

AC CURRENT (RMS)

– 4 ranges (3.3mA, 33mA, 330mA, 2.5A)
- Accuracy is better than 0.9% (20 Hz to 50KHz)

RESISTANCE (TWO WIRE)

- 8 ranges (33Ω, 330Ω, 3.3 KΩ, 33 KΩ, 330 KΩ, 3.3 MΩ, 33 MΩ, 330 MΩ)
- Accuracy is better than 0.003%

RESISTANCE (FOUR WIRE)

- 8 ranges (33Ω,330Ω, 3.3 KΩ, 33 KΩ, 330 KΩ, 3.3 MΩ, 33 MΩ, 330 MΩ)
- Accuracy is better than 0.003%

RESISTANCE (SIX WIRE GUARDED)

- 5 ranges (33Ω, 330Ω, 3.3 KΩ, 33 KΩ, 330 KΩ)
- Accuracy is better than 0.03%

TEMPERATURE

- Range is –150 degrees C to 650 degrees C.
- Accuracy and Range depend on RTD Type.
- The internal temperature can also be monitored

LEAKAGE

- 3 Ranges (100 nA, 1000nA, 3300nA)
- Accuracy is better than 2.0%

CAPACITANCE

- 7 Ranges (10nF, 100nF, 1uF, 10uF, 100uF, 1mF, 10mF)
- Accuracy is better than 2% .

INDUCTANCE

- 6 Ranges (33uH, 330uH, 3.3mH, 33mH, 330mH, 3.3H)

PEAK to PEAK, CREST, and MEDIAN

- 4 Ranges (330mV, 3.3V, 33V, 250V)
– Accuracy is better than 1.5%

TIMING FUNCTIONS -5 Timing Functions are available ACV Frequency, ACI Frequency, Duty Cycle, Pulse Width, & Totalizer.

AC/DC VOLTAGE SOURCE AND CURRENT SOURCE

- DC VOLTAGE SOURCE : (-10.0V TO +10.0v). Accuracy is better than 0.13%)
- AC VOLTAGE SOURCE : (0 to 20 V Peak to Peak) Accuracy is better than 0.8%.
- DC CURRENT SOURCE : 5 Ranges (1.25uA, 12.5uA, 125uA, 1.25mA, 12.5mA) Accuracy is better than 0.2%

Mechanical:

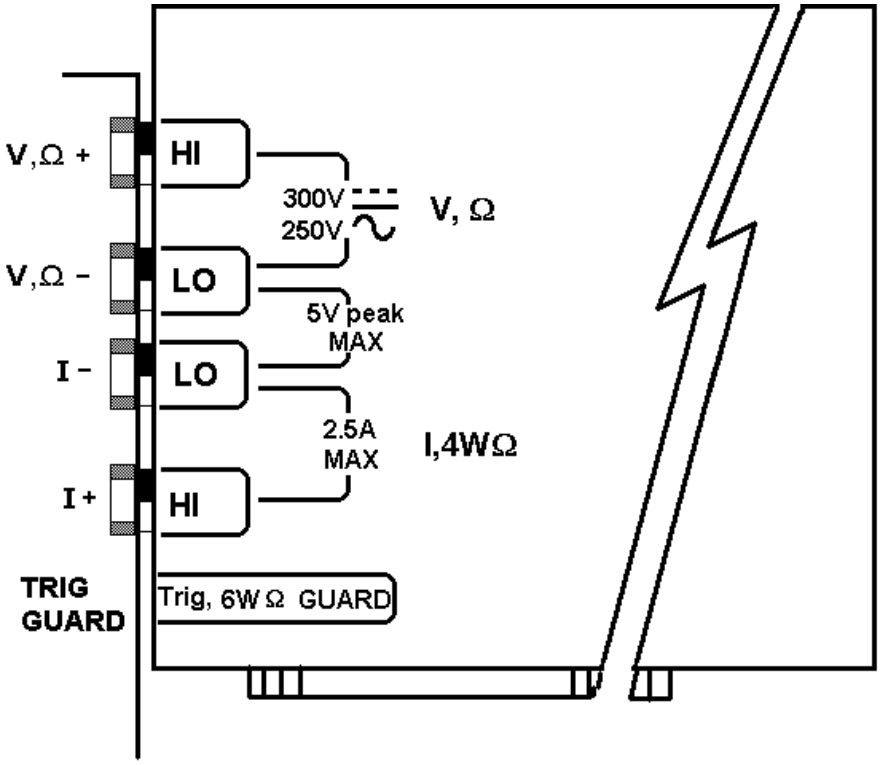
Thickness:	1.200 inches
Width:	10.317 inches
Length:	13.78 inches
Weight:	3 lbs.

Environmental Specifications

Temperature:	
Operating:	0º to 55ºC
Storage:	- 40º to 75ºC
Relative Humidity:	
Operating:	0 to 90% non-condensing
Storage:	0 to 95% non-condensing

Chapter 8 DMM Input Connectors

Before using the DMM, please take a few moments and review this section to understand where the voltage, current, or resistance and other inputs and outputs should be applied. **This section contains important information concerning voltage and current limits. Do not exceed these limits, as personal injury or damage to the instrument, your computer or application may result.**



The DMM input connectors. **(TRIG GUARD with Model 1005 only)**

V, Ω + This is the positive terminal for all Volts, 2WΩ, capacitance, diode and inductance measurements, and for sourcing of VDC, VAC and IDC. It is also the Source HI for 4WΩ measurements. The maximum input across **V, Ω +** and **V, Ω -** is 300 VDC or 250 VAC when in the measuring mode**. When in the sourcing mode, the maximum input allowed before damage occurs is 100 volts.**

V, Ω - This is the negative terminal for all Volts, 2WΩ, capacitance diode and inductance measurements, and or sourcing of VDC, VAC and IDC. It is also the Source LO for 4WΩ. **Do not float this terminal or any other DMM terminal more than 300 VDC or 250 VAC above Earth Ground.** (Also, see **Trig, 6W Guard** below.)

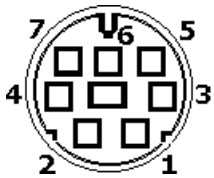
I + This is the positive terminal for all Current measurements. It is also the Sense HI for 4WΩ measurements and 6WΩ guarded measurements. The maximum input across **I, 4WΩ +** and **I, 4WΩ -** is **2.5 A**. Do not apply more than 5 V peak across these two terminals!

I – This is the negative terminal for all Current measurements. In the Current modes, it is protected with a **2.5 A, 250 V Fast Blow fuse** (5 x 20 mm). It is also the Sense LO for 4WΩ measurements and 6WΩ guarded measurements. **V, Ω -** and **I, 4WΩ -** should never have more than 5 V peak across them.

TRIG GUARD (Model 1005 only) Both the Trigger and Guard functions use the DIN-7 connector. This group of pins include the positive and negative hardware trigger input lines and the two MODEL 1005 Guarded Measurement Force and Sense signals. The external trigger initiates reading(s) into the onboard buffer, and the 6W guard signals facilitate in-circuit resistor measurements by means of isolating a loading node. The DIN-7 plug can be ordered from Ascor and is also available at many electronic hardware distributors. The connector is generically referred to as a mini DIN-7 male. The trigger signal should be in the range of 3 V to 12 V peak. The two 6W guard signals should never have more than 5 V peak across them.

Warning! The DIN connector pins are protected to a maximum of 35 V with respect to the PC chassis and any other DMM terminal. Do not apply any voltages greater than 35 V to the DIN connector pins. Violating this limit may result in personal injury and/or permanent damage to the DMM.

DIN-7, Pin number	Function
7	External Trigger, Positive terminal
4	External Trigger, Negative terminal
1	Guard Source (MODEL 1005)
6	Guard Sense (MODEL 1005)



DIN-7 Connector Pin Description, view from bracket side.

Chapter 9 README FILE FOR 38xx DMM

THE FOLLOWING “README” FILE IS PRESENTED AS A REFERENCE DOCUMENT.

It contains valuable information concerning installation procedures, VXI soft front panel operation, sample programs, and troubleshooting information. The information contained was complete as of the date and revision noted on the “readme” document. HOWEVER you should always consult the “readme” file that comes with the software driver (readme.txt) for the most current information.

NOTE

The *Plug&Play* software driver for the 38XX DMM must be at revision level 2.0.0 or higher in order for the DMM to be compatible with the Model GT-8300A LAN VXI Enclosure.

```
        ---------------------------------------------
                          README for
                    ASCOR DMM Module Carrier
                   VXIplug&play driver
                    for Windows
                     Revision 1.0.2
                          January 2004
        ---------------------------------------------

      (c) Copyright ASCOR, Inc. 2004. All rights reserved.

This document contains the ASCOR VXIplug&play Instrument Driver information
for ASCOR 3801 DMM Module Carrier.

The audience of this document are the following:
  (A) Users who have our VXIplug&play distribution floppy disks.
  (B) Users who obtained the 3801.zip web package via our website
            or received the 3801.zip web package via your email
  (C) Users who received disk(n).zip distribution floppy disk image files
  (D) Users who already have installed the driver.

--------
CONTENTS
--------
Revision History
Update Notes
Requirements
List Of Files
Installation Procedure
Resource Manager
VXI Soft Front Panel
Sample Programs
Sample Programs Instruction
Required Files for Compiling, Linking, and Execution
Known Issues
Troubleshooting
```

```
Contact Information

REVISION HISTORY
================

    Revision 1.0.3: Modified function as1005_reset() to set function mode to
                    VDC with the range 330V, took off carrier reseting part,
                    and set auto-range mode off.
                    Added as3801_getMainFrameID()

    Revision 1.0.2: Modified function and control helps in as1005.fp
                Added as1005_getMainFrameID()
                Renamed all as1005 functions with DMM after prefix as1005_
                in as1005.h and as1005.c files.

    Revision 1.0.1: Driver is modified to save the DMM number.
                Began calling as1005 functions from this rev.
                Replaced as1005_init function with
                as_1005dmmInit function.
                Replaced as1005_close function with
                as_1005terminate function.

    Revision 1.0.0: This is the original revision of ASCOR
                3801/3802 DMM.n.

UPDATE NOTES
==============

    If you have an older version of the VXIplug&play driver
    in your system and want to update:
        Make sure that you have the new version available.
        Uninstall the previous version.
        Install the new version of the VXIplug&play driver.

    You can download new and updated version of the
    VXIplug&play driver from ASCOR web site at
    http://www.ascor.com/downloads.htm#drivers


REQUIREMENTS
============

    VXIplug&play driver requires the VISA Library (VISA32.DLL) to be resident
    in the \WINDOWS\SYSTEM directory on the system controller for WIN95/WIN98
    or in the \WINDOWS\SYSTEM32 directory for WINNT/WIN2000.  The software
    included with this module requires VISA Version 1.0 or higher.  If you
    have an older version of VISA for your embedded controllers or VXI
    interfaces, please contact your system controller vendor(s)
    for upgrade kits.


LIST OF FILES
=============

Web Package
```

```
-----------

    The 3801.zip web package consists of the following files.

        disk1.zip       First VXIplug&play Distribution Diskette zip file
        disk2.zip       Second VXIplug&play Distribution Diskette zip file
        vppCover.doc    ASCOR cover letter
        vppQA.doc       ASCOR VXIplug&play driver questions and answers
        readme.txt      This readme file


Distribution Diskettes
----------------------

    Diskette 1 of 2

        as3801.001      First compressed data file for setup.exe
        ascor_c.bmp     Background file for setup.exe
        ascor_m.bmp     Background file for setup.exe
        setup.exe       Installation program for this VXIplug&play driver
        readme.txt      This readme file

    Diskette 2 of 2

        as3801.002      Second compressed data file for setup.exe

    If the web package or the distribution diskettes do not contain all of
the
    above files, or appears to be damaged, please contact ASCOR, Inc.

INSTALLATION PROCEDURE
======================
    Follow the appropriate installation procedure based on the audience list
    found at the beginning of this document.
    Users (A)
    Using the VXIplug&play distribution floppy disks:
        Insert disk1 to the floppy drive.
        From the Taskbar Start menu select the Run option.
        Enter [drive]:\setup.exe.
        Press enter.
        Follow the instructions from the installation program.

    Users (B) and (C)
    From the Web Package or from distribution floppy disk image files:
        You have to unzip Web Package to have zip files.

        Please locate two zip files: disk1.zip and disk2.zip.  There are two
        choices for installing this VXIplug&play driver.
        From the hard disk or from the floppy disks.

            Hard Disk:
                In Windows 95, 98, NT, or 2000, create a temporary directory
                if you do not already have one.
                Unzip both disk1.zip and disk2.zip onto the temporary
                directory.
```

```
            From the temporary directory run setup.exe file.
            Follow the instructions from the installation program.
            The files copied onto the temporary directory can be deleted
            after the installation process is completed.

        Floppy Disks:
            Requires two 3.5 inch formatted disks.
            Create disk1 by unzipping files from disk1.zip.
            Create disk2 by unzipping files from disk2.zip.
            Insert disk1 to the floppy drive.
            From the Taskbar Start menu select the Run option.
            Enter [drive]:\setup.exe.
            Press enter.
            Follow the instructions from the installation program.
```

RESOURCE MANAGER

```
================
    A Resource Manager (Resman) is a utility defined by the VXIbus
    specification.  Its main tasks are to identify all VXI devices in
    mainframe(s), allocates logical addresses to those devices in
    mainframe(s), and then make sure that the devices have passed their
    self-tests.

    You need to run Resman if you are using MXI-VXI controller.
    You do not need to run Resman with GPIB-VXI controller.

    If you are using National Instruments Software:

    Resman.exe is located in ...\National Instruments\Vxi folder.
    (... is the directory where you install VXI; C:\Program Files
    is default directory.)

    And you can run Resman from:
    Start >> Programs >> National Instruments >> Vxi >> Resman
```

VXI SOFT FRONT PANEL

```
====================
    Soft front panel program provides interactive session between end-user
and
    the instrument.  Soft front panel program uses graphical user interface
    technology to present the equivalent of knobs, buttons, and controls.
The
    user manipulates these controls with a mouse or with the computer
    keyboard. Soft front panel program introduces the instrument and gives a
    demonstration of its capability.  Depending on the complexity of the soft
    front panel, all of instrument driver commands may be exercised, making
    the soft front panel an excellent driver testing tool as well.

    To run ASCOR 3801 Front Panel:
    Start >> Programs >> Vxipnp >> as3801 Front Panel
```

SAMPLE PROGRAM

```
==============
    ASCOR provides two sample programs along with ASCOR 3801/3802 driver.
```

These sample programs demonstrate how to control ASCOR DMM 1004/1005 using
our ASCOR 3801/380 VXIplug&play Instrument Driver. These programs are installed as part of the ASCOR 3801/3802 VXIplug&play Instrument Driver.

Before running the sample program, the user must first set the logical address of the DMM to 2 and run the Resource Manager program (see Resource Manager).

sampleVDC.exe:
This sample program demonstrates how to measure DC Voltage with ASCOR DMM. This sample program displays a sign on message and proceeds to initialize the the DMM. After a successful initialization the program prompts the user to type a selection character followed by an Enter key. The available selection characters are "M" for DC Voltage measurement and "Q" for exiting the program.

sampleMeasurement.exe:
This sample program displays a sign on message and proceeds to initialize the DMM. After a successful initialization the program shows the selection list and prompts the user to type a selection character followed by an Enter key. The available selection characters are "A" for selecting VAC mode, "D" for selecting VDC mode, "F" for selecting Frequency mode, "P" for selecting Period mode, "R" for selecting 2-wire Resistance mode, and "Q" for exiting the program.

If you select any of selection characters for selecting mode operation, excepting character "Q", then the program again prompts the user to type a selection character followed by an Enter key. The available selection characters are "M" for measurement, "S" for stopping and going back to the selection list, and "Q" for exiting the program.

These sample programs are located in C:\Vxipnp\Winxx\as3801 folder.
Note: For Win9x Operating System substitute Winxx with WIN95.
For WinNT and Win2000 Operating System substitute Winxx with WINNT.

Sample C code files, sampleVDC.c and sampleMeasurement.c, can be modified and compiled on any C language development environment. They are located in the same folder with the sample programs, in C:\Vxipnp\Winxx\as3801 folder.
The function prototypes for the VXIplug&play driver functions used in the sample programs can be found in the include files, as1005.h and as3801.h. These files are part of the ASCOR 3801/3802 VXIplug&play driver and they are located in C:\Vxipnp\Winxx\include folder.

SAMPLE PROGRAMS INSTRUCTION

==========

To run these programs:
- Locate these programs in C:\Vxipnp\Winxx\as3801 folder.
- Select the program you wish to run.
- Press Enter.

Following is a step by step instruction of the sample programs.

sampleVDC.exe:

```
* Type M and press the Enter to measure DC Voltage.

* Type Q and press the Enter to exit the program.

sampleMeasurement.exe:

* Type A and press the Enter to select VAC mode.

  + Type M and press the Enter to measure AC Voltage.

  + Type S and press the Enter to stop and return to the selection list.

  + Type Q and press the Enter to exit the program.

 * Type D and press the Enter to select VDC mode.

  + Type M and press the Enter to measure DC Voltage.

  + Type S and press the Enter to stop and return to the selection list.

  + Type Q and press the Enter to exit the program.

* Type F and press the Enter to select Frequency mode.

  + Type M and press the Enter to measure Frequency.

  + Type S and press the Enter to stop and return to the selection list.

  + Type Q and press the Enter to exit the program.

* Type P and press the Enter to select Period mode.

  + Type M and press the Enter to measure Period.

  + Type S and press the Enter to stop and return to the selection list.

  + Type Q and press the Enter to exit the program.

* Type R and press the Enter to select 2-wire Resistance mode.

  + Type M and press the Enter to measure 2-wire Resistance.

  + Type S and press the Enter to stop and return to the selection list.

  + Type Q and press the Enter to exit the program.

* Type Q and press the Enter to exit the program.
```

REQUIRED FILES FOR COMPILING, LINKING, AND EXECUTION

==========

```
ASCOR provides files to help users who wish to create their own programs
for controlling ASCOR DMM 1004/1005.  Different files are used for
```

```
    compiling, linking, and execution.
    * The following header files are required for compiling: as3801.h and
      as1005.h.
    * The following library files are required for linking: as3801_32.lib and
      as1005.lib.
    * The following dynamic link library files are required for execution:
      as3801_32.dll, smv2040.dll, asXMod_32.dll, and as1005_32.dll.

    These files are installed automatically during the ASCOR 3801
VXIplug&play
    driver installation. These files are located in different folders for
    different frameworks (WIN 95 and WIN NT).

    WIN 95 Framework
    ----------------

    * Compiling
        + C:\vxipnp\Win95\include\as3801.h
        + C:\vxipnp\Win95\include\as1005.h

    * Linking (See Note Below)
        + C:\vxipnp\Win95\lib\msc\as3801_32.lib
        + C:\vxipnp\Win95\lib\msc\as1005.lib

    * Execution
        + C:\vxipnp\Win95\bin\as3801_32.dll
        + C:\Windows\System\smv2040.dll
        + C:\vxipnp\Win95\bin\asXMod_32.dll
        + C:\vxipnp\Win95\bin\as1005_32.dll


    WIN NT Framework
    ----------------

    * Compiling
        + C:\vxipnp\WinNT\include\as3801.h
        + C:\vxipnp\WinNT\include\as1005.h

    * Linking (See Note Below)
        + C:\vxipnp\WinNT\lib\msc\as3801_32.lib
        + C:\vxipnp\WinNT\lib\msc\as1005.lib

    * Execution
        + C:\vxipnp\WinNT\bin\as3801_32.dll
        + C:\Windows\System32\smv2040.dll
        + C:\vxipnp\WinNT\bin\asXMod_32.dll
        + C:\vxipnp\WinNT\bin\as1005_32.dll


    Note: For Borland builder substitute \msc\ with \bc\ folder
          For Symantec builder substitute \msc\ with \sc\ folder
          For Watcom builder substitute \msc\ with \wc\ folder
```

KNOWN ISSUES
==========

ASCOR VXIplug&play 3801 driver supplies a function called autoConnectToAll. This function makes connection to all the ASCOR 3801 instruments automatically and resets these instruments as well.

Since this function will reset all ASCOR 3801 instruments, the user should use it only one if he/she does not want to reset all ASCOR 3801 instruments again.

TROUBLESHOOTING
==========

Problem: Project link error
Undefined symbol ' as1005_dmmInit@…' referenced in "xxx.c".
(xxx.c: is your c file)

Cause: User used the old as1005.lib which does not have function as1005_dmmInit() when linking to xxx.c

Solution: Make sure all library files, as1005.lib and as3801_32.lib, are up-to-date. The latest library files, as1005.lib and as3801_32.lib, are located in C:\Vxipnp\Win95\Lib\yyyy folder after ASCOR 3801/3802 VXIplug&play Instrument driver is installed.
(yyyy: the library folder which is suitable for the user compiler)

Problem: "DMM Uninitialized" error appears when attempting to load as3801 Front Panel

Cause: as3801 Soft Front Panel does not see DMM.

Solution: Run Resman

Problem: "Fail W/R to Comm. Controller" error appears when attempting to load as3801 Front Panel.

Cause: Fail to Write/Read to/from DMM.

Solution: Turn on chassis with DMM inside and let it runs 2 minutes for warming up. Run Resman and then run as3801 Front Panel.

Problem: "Over Range" error appears when attempting to measure.

Cause: The source for measurement is out of range from the range the user sets.

Solution: Select appropriate range or set autoRange on.

CONTACT INFORMATION
==========

If you have any questions, comments, or suggestions, we can be reached at the addresses below.

Web site: www.ascor-inc.com
E-mail: info@ascor-inc.com

Postal mail: ASCOR, Inc.
4384 Enterprise Place,
Fremont, CA 94538 USA
Voice: 510-490-2300 Fax: 510-490-8493

Chapter 10 SAMPLE PROGRAMS

The following two sample programs are examples of how to program the 38XX DMM. The examples are the actual source code written in C.

DC VOLTAGE MEASUREMENT PROGRAM

This program shows how to make a DC Volts measurement.

MEASURE BASIC MODES ON THE ASCOR DMM

This program interrogates the basic modes of the DMM such as AC/DC Voltage, Frequency, Period, Resistance, etc.

10.1 Sample – Dc Voltage Measurement Program

```
/*=============================================================================*/
/* Sample DC Voltage Measurement Program.
           */
/*=============================================================================*/
/* Copyright (C) 2003 ASCOR, Inc.
*/
/*=============================================================================*/
/* Program Revision:  1.0.0
                            */
/*=============================================================================*/
/* Modification History:                                                       */
/*
                                        */
/*
                                        */
/*=============================================================================*/

/*  Include files  */
#include <stdio.h>
#include <ctype.h>
#include "../include/as1005.h"
#include "../include/as3801.h"

ViSession Handle;
/* This program works if the resourceName and the DMM carrier have the same logical
address */
/* Set the logical address in the resourceName to reflect the setting on the DMM
carrier */
ViRsrc resourceName        = "vxi::2::instr";         //"vxi::logical address::instr"

int main (int argc, char *argv[])
{
       double  valueVDC;
       char   userInput[100];
       int    getVDC;


       printf("\nCopyright (C) 2003 ASCOR, Inc.\n");
```

```
      printf("Program Revision:  1.0.0\n\n");
      printf("This is a sample program to measure DC Voltage on ASCOR DMM
1004/1005.\n");
      /* This program works if the resourceName has the same logical address as the
DMM carrier */
      printf("This program works with the ASCOR 3801/3802 DMM carrier set to\n"
                  "logical address 2.\n\n");
      printf("Connection:\n");
      printf("     Connect positive source of DC Voltage to ASCOR DMM V+ terminal
and\n");
      printf("     Connect negative source of DC Voltage to ASCOR DMM V-
terminal.\n\n\n");
```

SAMPLE – DC VOLTAGE MEASUREMENT PROGRAM

```
      printf("     Initializing ASCOR DMM...");
      /* Initialize carrier      */
      as3801_init (resourceName, 1, 1, &Handle);
      /* Initialize ASCOR DMM    */


      as1005_DMMInit (Handle);
      printf("Initialized\n\n");

      /* Set function mode to DC Voltage      */
      as1005_DMMSetFunction(Handle, as1005_VDC);
      /* Set auto range          */
      as1005_DMMSetAutoRange(Handle, 1);

      /* Measure DC Voltage      */
      getVDC = 1;
      while(getVDC){
            printf( "Type M and <Enter> to measure DC Voltage\n"
                        "Type Q and <Enter> to quit the program\n");
            scanf("%s", userInput);

            if(toupper(userInput[0]) != 'Q'){
                  as1005_DMMReadNorm(Handle, &valueVDC);
                  printf("The DC Voltage value is %f\n\n", valueVDC);
            }
            else
                  getVDC = 0;
      }

      /* Terminate DMM    */
      as1005_DMMTerminate(Handle);
      /* Close carrier    */
      as3801_close (Handle);

      return 0;
}
```

10.2 Sample Measurement Program

```
/*=============================================================================*/
/* Sample Measurement Program.
             */
/*=============================================================================*/
/* Copyright (C) 2003 ASCOR, Inc.
*/
/*=============================================================================*/
/* Program Revision:  1.0.0
                          */
/*=============================================================================*/
/* Modification History:                                                       */
/*
                                        */
/*
                                        */
/*=============================================================================*/

/*  Include files  */
#include <stdio.h>
#include <ctype.h>
#include "../include/as1005.h"
#include "../include/as3801.h"

#define as1005_FREQ        as1005_VAC
#define as1005_PER         as1005_VAC


ViSession Handle;
/* This program works if the resourceName and the DMM carrier have the same logical
address */
/* Set the logical address in the resourceName to reflect the setting on the DMM
carrier */
ViRsrc resourceName        = "vxi::2::instr";         /*     "vxi::logical
address::instr"            */

int main (int argc, char *argv[])
{
       double  readMeasurement;
       char   userChar, userInput[100], functMode = 'Q';
       int    exitProg = 0, modeMeas;


       printf("\nCopyright (C) 2003 ASCOR, Inc.\n");
       printf("Program Revision:  1.0.0\n\n");
       printf("This is a sample program to measure basic modes on ASCOR DMM
1004/1005.\n");
       /* This program works if the resourceName has the same logical address as the
DMM carrier */
       printf("This program works with the ASCOR 3801/3802 DMM carrier set to\n"
                     "logical address 2.\n\n\n");


printf("     Initializing ASCOR DMM...");
       /* Initialize carrier      */
       as3801_init (resourceName, 1, 1, &Handle);
       /* Initialize ASCOR DMM    */
       as1005_DMMInit (Handle);
       printf("Initialized\n");
```

```
	/* Set auto range          */
	as1005_DMMSetAutoRange(Handle, 1);

	while(!exitProg){
		printf( "\n\nSelection list:\n"
				"Type A and <Enter> to select AC Voltage mode\n"
				"Type D and <Enter> to select DC Voltage mode\n"
				"Type F and <Enter> to select Frequency mode\n"
				"Type P and <Enter> to select Period mode\n"
				"Type R and <Enter> to select 2-wire Resistance mode\n"
				"Type Q and <Enter> to quit the program\n\n");
		scanf("%s", userInput);
		functMode = toupper(userInput[0]);

		if((functMode == 'A') || (functMode == 'D') || (functMode == 'F') ||
				(functMode == 'P') || (functMode == 'R')){

			switch(functMode){
				case 'A':
					/* Set ASCOR DMM to AC Voltage mode            */
					as1005_DMMSetFunction(Handle, as1005_VAC);
					printf("\nYou selected AC Voltage mode.\n");
					/* Connection */
					printf("\nConnection:\n");
					printf("    Connect positive source of AC Voltage to ASCOR DMM V+
terminal and\n");
					printf("    Connect negative source of AC Voltage to ASCOR DMM V-
terminal.\n\n");

					modeMeas = 1;
					while(modeMeas){
						printf( "Type M and <Enter> to measure AC
Voltage\n"
								"Type S and <Enter> to stop and
go back to the selection list\n"
								"Type Q and <Enter> to quit the
program\n");
						scanf("%s", userInput);


						userChar = toupper(userInput[0]);

						if((userChar != 'S') && (userChar != 'Q')){
							/* Read measurement */
							as1005_DMMReadNorm(Handle,
&readMeasurement);
							printf("The AC Voltage value is %f
volts\n\n", readMeasurement);
						}
						else{
							modeMeas = 0;
							if(userChar == 'Q')
								exitProg = 1;
						}
					}
					break;



				case 'D':
					/* Set ASCOR DMM to DC Voltage mode            */
```

```
                                as1005_DMMSetFunction(Handle, as1005_VDC);
                                printf("\nYou selected DC Voltage mode.\n");
                                /* Connection */
                                printf("Connection:\n");
                                printf("    Connect positive source of DC Voltage to ASCOR DMM V+
terminal and\n");
                                printf("    Connect negative source of DC Voltage to ASCOR DMM V-
terminal.\n\n");

                                modeMeas = 1;
                                while(modeMeas){
                                      printf( "Type M and <Enter> to measure DC
Voltage\n"
                                                  "Type S and <Enter> to stop and
go back to the selection list\n"
                                                  "Type Q and <Enter> to quit the
program\n");
                                      scanf("%s", userInput);

                                      userChar = toupper(userInput[0]);

                                      if((userChar != 'S') && (userChar != 'Q')){
                                            /* Read measurement */
                                            as1005_DMMReadNorm(Handle,
&readMeasurement);
                                            printf("The DC Voltage value is %f
volts\n\n", readMeasurement);
                                      }


                                      else{
                                            modeMeas = 0;
                                            if(userChar == 'Q')
                                                  exitProg = 1;
                                      }
                                }
                                break;

                          case 'F':
                                /* Set ASCOR DMM to Frequency mode             */
                                as1005_DMMSetFunction(Handle, as1005_FREQ);
                                printf("\nYou selected Frequency mode.\n");
                                /* Connection */
                                printf("Connection:\n");
                                printf("    Connect positive source of AC Voltage to ASCOR DMM V+
terminal and\n");
                                printf("    Connect negative source of AC Voltage to ASCOR DMM V-
terminal.\n\n");

                                modeMeas = 1;
                                while(modeMeas){
                                      printf( "Type M and <Enter> to measure
Frequency\n"
                                                  "Type S and <Enter> to stop and
go back to the selection list\n"
                                                  "Type Q and <Enter> to quit the
program\n");

                                      scanf("%s", userInput);

                                      userChar = toupper(userInput[0]);

                                      if((userChar != 'S') && (userChar != 'Q')){
```

```
                                        /* Read measurement */
                                        as1005_DMMReadFrequency(Handle,
&readMeasurement);
                                        printf("The Frequency value is %f
Hz\n\n", readMeasurement);
                                }
                                else{
                                        modeMeas = 0;
                                        if(userChar == 'Q')
                                                exitProg = 1;
                                }
                        }
                        break;


                case 'P':
                        /* Set ASCOR DMM to Period mode          */
                        as1005_DMMSetFunction(Handle, as1005_PER);
                        printf("\nYou selected Period mode.\n");
                        /* Connection */
                        printf("Connection:\n");
                        printf("    Connect positive source of AC Voltage to ASCOR DMM V+
terminal and\n");
                        printf("    Connect negative source of AC Voltage to ASCOR DMM V-
terminal.\n\n");

                        modeMeas = 1;
                        while(modeMeas){
                                printf( "Type M and <Enter> to measure
Period\n"
                                        "Type S and <Enter> to stop and
go back to the selection list\n"
                                        "Type Q and <Enter> to quit the
program\n");
                                scanf("%s", userInput);

                                userChar = toupper(userInput[0]);

                                if((userChar != 'S') && (userChar != 'Q')){
                                        /* Read measurement */
                                        as1005_DMMReadPeriod(Handle,
&readMeasurement);
                                        printf("The Period value is %f
seconds\n\n", readMeasurement);
                                }
                                else{
                                        modeMeas = 0;
                                        if(userChar == 'Q')
                                                exitProg = 1;
                                }
                        }
                        break;

                case 'R':


                        /* Set ASCOR DMM to 2-wire Resistance mode
                        */
                        as1005_DMMSetFunction(Handle, as1005_OHMS2W);
                        printf("\nYou selected 2-wire resistance mode.\n");
                        /* Connection */
                        printf("Connection:\n");
```

```
                                    printf("     Connect one end of the resistor to ASCOR
DMM V+ terminal and\n");
                                    printf("     Connect the other end of the resistor to ASCOR DMM V-
terminal.\n\n");


                                    modeMeas = 1;
                                    while(modeMeas){
                                          printf( "Type M and <Enter> to measure 2-wire
resistance\n"
                                                      "Type S and <Enter> to stop and
go back to the selection list\n"
                                                      "Type Q and <Enter> to quit the
program\n");
                                          scanf("%s", userInput);

                                          userChar = toupper(userInput[0]);

                                          if((userChar != 'S') && (userChar != 'Q')){
                                                /* Read measurement */
                                                as1005_DMMReadNorm(Handle,
&readMeasurement);
                                                printf("The resistance value is %f
Ohms\n\n", readMeasurement);
                                           }
                                           else{
                                                modeMeas = 0;
                                                if(userChar == 'Q')
                                                      exitProg = 1;
                                          }
                                    }
                                    break;

                        }
                  }
                  else if(functMode == 'Q')
                        exitProg = 1;

      }

      /* Terminate DMM    */
      as1005_DMMTerminate(Handle);
      /* Close carrier    */
      as3801_close (Handle);

      return 0;
}
```

Chapter 11 **Aquiris DMM Manual**

TABLE OF CONTENTS

1.0 Introduction

Congratulations! You have purchased an instrument with analog and systems performance that rivals the best, all-in-one box, instruments. The MODEL 1004 and MODEL 1005 series digital multimeters (DMMs) are easy to setup and use, have sophisticated analog and digital circuitry to provide very repeatable measurements, and are protected to handle any unexpected situations your measurement environment may encounter. To get years of reliable service from these DMMs, please take a few moments and review this manual before installing and using this precision instrument.

This manual describes the MODEL 1004 and MODEL 1005 DMMs. Each DMM delivers unmatched breakthrough performance in a VXI instrument. With a rich repertoire of functions, the MODEL 1004 series out performs all other plug-in DMMs.

Note: In this manual, all references to the "MODEL 1004" apply to the Ascor MODEL 1004, and references to MODEL 1005 and DMM/LCR apply to the Ascor MODEL 1005 . The term "DMM" will frequently be used to reference both types of units. Features unique to the MODEL 1005 will be identified as such.

1.1 Safety Considerations

Safety Considerations

The MODEL 1004 series of DMMs is capable of measuring up to 300 VDC or 250 VAC across the Volt HI and LO terminals, and can also measure common mode signals that "float" the DMM above EARTH ground by up to 300 VDC or 250 VAC. When making common mode measurements, the majority of the circuits inside the DMM are at the common mode voltage. **These voltages can be lethal and can KILL! During and after installing your DMM, check to see that there are no wires or ribbon cables from your PC trapped inside the DMM.**

The DMM comes installed with four shields (bottom, top and two edge strips) that **must not be removed for performance as well as safety reasons.** Removal of these shields and/or improper assembly of the shields can result in lethal voltages occurring within your PC. Be sure to check your installation before closing the cover on your personal computer.

Warning

Check to see that no loose wires or ribbon cables infringe upon any of the internal circuits of the DMM, as this may apply measurement voltages to your computer, causing electrocution and/or damage to your computer!

To avoid shock hazard, install the DMM only into a computer that has its power connector connected to a power receptacle with an earth safety ground.

When making any measurements above 50 VDC or 40 VAC, only use Safety Test Leads. Examples of these are the Ascor Basic Test Leads and Deluxe Test Leads, offered as an accessory with the Ascor DMMs.

1.2 Minimum Requirements

The MODEL 1004 and the MODEL 1005 are meant to be used with Ascor's 38xx VXI series and other Ascor Inc. instrumentation units. A VXI *Plug & Play* driver is provided by Ascor for the 38XX VXI series

1.3 Feature Set

The base unit, the MODEL 1004, has 6-1/2 digit performance and can be used as a general purpose DMM, giving very accurate and stable readings. The state-of-the art MODEL 1005 is a superset of the MODEL 1004, adding inductance & capacitive measurement and sourcing capabilities.

MODEL 1004 and MODEL 1005 6½ Digit DMMs feature table:

Function	**MODEL 1004 DMM**	**MODEL 1005 LCR Sourcing DMM**
DCV 4 ranges, >10 GΩ & 10 MΩ input resistance.	√	√
ACV 4 ranges, 1 MΩ input	√	√
2-Wire Ohms, six ranges 330 Ω to 33 MΩ	√	***plus*** 33 Ω, 330 MΩ
4-Wire Ohms, four ranges 330 Ω to 330 kΩ	√	***plus*** 33 Ω range
DC current, four ranges 3.3 mA to 2.5 A	√	√
AC current, four ranges 3.3 mA to 2.5 A	√	√
Diode V/I characteristics at 100 ηA to 1mA	√	***plus*** 10 mA
Auto range, Relative	√	√
Min/Max, dB and percent deviation functions	√	√
On board measurement buffer	√	√
Measurement rate: 0.2 to 1,000/sec	√	√
External and threshold trigger	√	√
Capacitance, seven ranges, 10 nF to 10 mF		√
Temperature (five basic RTD types)		√
Frequency / Period measurement		√
Pulse width, pos./neg., & duty cycle		√

Totalizer/event counter		√
Variable threshold DAC; all timing measure.		√
Peak to Peak, Crest factor, Median		√
Internal DMM temperature sensor		√
Six wire Ohms (with force/sense)		√
Inductance, six ranges 33 µH to 3.3 H		√
DCV source 0 to +/-10.0 V		√
ACV source 0 to 20 V pk-pk, 2 Hz to 75 KHz		√
DC current source, 1 nA to 12.5 mA		√
Leakage (with external 1Meg)		√
High Ohms range 1,000 Meg		√

2.0 Specifications

2.1 DC Voltage Measurement

Input Characteristics

Input Resistance 330 mV & 3.3 V Ranges: >10 GΩ ,

Input Resistance 33 V & 330 V Ranges: 10 MΩ

Accuracy ± (% of reading + Volts) [1]

Range	Full Scale 6 ½ Digits	Resolution	24 hours 23°C ± 1°C	90 Days 23°C ± 5°C	One Year 23°C ± 5°C
330 mV	330.0000 mV	100 ηV	0.003 + 4.5 µV	0.004 + 5.5 µV	0.007 + 8 µV
3.3 V	3.300000 V	1 µV	0.002 + 10 µV	0.0025 + 12 µV	0.0045 + 17 µV
33 V	33.00000 V	10 µV	0.003 + 250 µV	0.004 + 280 µV	0.007 + 330 µV
330 V	330.0000 V	100 µV	0.004 + 1 mV	0.005 + 1.2 mV	0.008 + 1.5 mV

[1] With reading rate set to two readings per second (rps) or slower, and within one hour of DCV zero, using Relative control.

DCV Noise Rejection Normal Mode Rejection, at 50, 60, or 400 Hz ± 0.5%, is better than 95 dB for reading rates of 10 rps or lower. Common Mode Rejection (with 1 kΩ lead imbalance) is better than 120 dB for these conditions.

2.2 DC Current Measurement

Input Characteristics

Burden Voltage < 350 mV for all ranges

Protected with 2.5A fuse (5x20mm, 250 V Fast)

Accuracy ± (% of reading + Amps) [1]

Range	Full Scale 5 ½ Digits	Resolution	24 hours 23°C ± 5°C	90 Days 23°C ± 5°C	One Year 23°C ± 5°C
3.3 mA	3.30000 mA	10 ηA	0.052 + 200 ηA	0.07 + 350 ηA	0.1 + 400 ηA
33 mA	33.0000 mA	100 ηA	0.04 + 1 µA	0.06 + 2 µA	0.1 + 3 µA
330 mA	330.000 mA	1 µA	0.05 + 30 µA	0.055 + 40 µA	0.075 + 60 µA
2.5 A	2.50000 A	10 µA	0.55 + 50 µA	0.6 + 200 µA	0.65 + 350 µA

[1] With reading rate set to 2 rps or slower, and within one hour of DCI zero, using Relative control.

2.3 AC Voltage Measurements

Input Characteristics

Input Resistance 1 MΩ, shunted by < 100 pF, all ranges

Crest Factor 3 at Full Scale, increasing to 7 at Lowest Specified Voltage

AC coupled Specified range: 10 Hz to 100 kHz

Typical Settling time < 0.5 sec to within 0.1% of final value

2.3.1 AC Voltage True RMS Measurement

Range	Full Scale 6 ½ Digits	Lowest specified Voltage	Resolution
330 mV	330.0000 mV	5 mV [1]	100 ηV
3.3 V	3.300000 V	10 mV	1 μV
33 V	33.00000 V	100 mV	10 μV
250 V [2]	250.0000 V	1 V	100 μV

[1] Between 5 mV and 10 mV, add 100 μV additional error to the accuracy table below. In many computer installations, if the DMM is not near a noisy board, usable voltage measurements of 1 mV can be obtained.

[2] Signal is limited to $8x10^6$ Volt Hz Product. For example, the largest frequency input at 250 V is 32 kHz, or $8x10^6$ Volt x Hz.

Accuracy ± (% of reading + Volts) [1]

Range	Frequency	24 hours 23°C ± 1°C	90 Days 23°C ± 5°C	One Year 23°C ± 5°C
330 mV	10 Hz - 20 Hz	3.0 + 350 μV	3.1 + 380 μV	3.2 + 430 μV
	20 Hz - 47 Hz	0.92 + 150 μV	0.93 + 170 μV	0.95 + 200 μV
	47 Hz - 10 kHz	0.13 + 100 μV	0.14 + 110 μV	0.15 + 120 μV
	10 kHz - 50 kHz	0.55 + 160 μV	0.6 + 200 μV	0.63 + 230 μV
	50 kHz - 100 kHz	5.3 + 350 μV	5.4 + 370 μV	5.6 + 400 μV
3.3 V	10 Hz - 20 Hz	3.0 + 2 mV	3.1 + 2.2 mV	3.2 + 2.5 mV
	20 Hz - 47 Hz	0.93 + 1.3 mV	0.96 + 1.5 mV	1.0 + 1.7 mV
	47 Hz - 10 kHz	0.05 + 1 mV	0.055 + 1.1 mV	0.065 + 1.2 mV
	10 kHz - 50 kHz	0.62 + 1.2 mV	0.65 + 1.3 mV	0.70 + 1.5 mV
	50 kHz - 100 kHz	5.1 + 1.5 mV	5.2 + 1.7 mV	5.3 + 2 mV
33 V	10 Hz - 20 Hz	3.0 + 14 mV	3.1 + 16 mV	3.3 + 20 mV
	20 Hz - 47 Hz	0.93 + 12 mV	0.96 + 14 mV	1.0 + 16 mV

	47 Hz - 10 kHz	0.06 + 10 mV	0.065 + 11 mV	0.073 + 13 mV
	10 kHz - 50 kHz	0.31 + 18 mV	0.33 + 21 mV	0.35 + 25 mV
	50 kHz - 100 kHz	2.0 + 30 mV	2.2 + 35 mV	2.4 + 40 mV
250 V	10 Hz - 20 Hz	3.0 + 140 mV	3.1 + 160 mV	3.3 + 200 mV
	20 Hz - 47 Hz	0.93 + 120 mV	0.96 + 130 mV	1.0 + 150 mV
	47 Hz - 10 kHz	0.04 + 100 mV	0.045 + 110 mV	0.06 + 130 mV
	10 kHz - 50 kHz	0.32 + 150 mV	0.4 + 170 mV	0.45 + 200 mV
	50 kHz - 100 kHz	2.5 + 200 mV	2.8 + 240 mV	3.2 + 300 mV

ACV Noise Rejection Common Mode rejection, for 50 Hz or 60 Hz with 1 kΩ imbalance in either lead, is better than 60 dB.

2.3.2 AC Peak-to-Peak Measurement (MODEL 1005)

Measures the peak-to-peak value of a repetitive waveform

ACV Range	Lowest specified input voltage (Vp-p)	Full Scale reading (Vp-p)	Resolution	Typical Accuracy 23°C ± 5°C One Year [1]
330 mV	0.1 V	1.85 V	1 mV	1.5 ±10 mV
3.3 V	1.0 V	18.5 V	10 mV	1.4 ±70 mV
33 V	10 V	185.0 V	100 mV	1.0 ±700 mV
250 V	100 V	850.0 V	1 V	1.0 ± 6 V

[1] Specified from 30Hz to 10 kHz. Input signal frequency of 30 Hz to 30 kHz.

2.3.3 AC Crest Factor Measurement (MODEL 1005)

Measures the crest factor (peak / RMS) of a repetitive waveform

ACV Range	Lowest specified input voltage (Vp-p)	Highest specified input voltages (Vp-p)	Resolution	Typical Accuracy 23°C ± 5°C One Year [1]
330 mV	0.1 V	1.8 V	0.01	2.2 ±0.3
3.3 V	1.0 V	18 V	0.01	2.1 ±0.1
33 V	10 V	180 V	0.01	2.0 ±0.1
250 V	100 V	700 V	0.01	2.0 ±0.1

[1] Crest factor measurement requires signal frequency of 30 Hz to 30 kHz.

2.3.4 AC Median Value Measurement (MODEL 1005)

Measures the mid-point between the positive and negative peaks of a repetitive waveform

Used to determine the Threshold DAC setting for optimal frequency and timing measurements

ACV Range	Lowest specified input voltage (Vp-p)	Full Scale reading	Resolution	Typical Accuracy 23°C ± 5°C One Year [1]
330 mV	0.08 V	±0.95 V	1 mV	2.0% ±17 mV
3.3 V	0.80 V	±9.5 V	10 mV	3% ±160 mV

33 V	8 V	±95.0 V	100 mV	3% ±1.4 V
250 V	80 V	±350.0 V	1 V	3% ±12 V

[1] Median measurements require a repetitive signal with frequency range of 30 Hz to 30 KHz.

2.4 AC Current Measurement, True RMS

Input Characteristics

Burden Voltage < 350 mV RMS all Ranges

Crest Factor 3 at Full Scale, increasing to 7 at Lowest Specified Current

Protected with 2.5 A fuse (5x20 mm, 250 V Fast)

Range	Full Scale 6 1/2 Digits	Lowest Specified Current	Resolution
3.3 mA	3.300000 mA	50 µA	1 nA
33 mA	33.00000 mA	500 µA	10 nA
330 mA	330.0000 mA	5 mA	100 nA
2.5 A	2.500000 A	50 mA	1 µA

Accuracy ± (% of reading + Amps)

Range	Frequency	24 hours 23°C ± 1°C	90 Days 23°C ± 10°C	One Year 23°C ± 10°C
3.3 mA	10 Hz - 20 Hz	3.8 + 4 µA	2.7 + 4 µA	2.9 + 4 µA
	20 Hz - 47 Hz	0.9 + 4 µA	0.9 + 4 µA	1.0 + 4 µA
	47 Hz - 1 kHz	0.04 + 1.5 µA	0.08 + 3 µA	0.12 + 4 µA
	1 kHz - 10 kHz	0.12 + 4 µA	0.14 + 4 µA	0.22 + 4 µA
33 mA	10 Hz - 20 Hz	1.8 + 30 µA	2.6 + 30 µA	2.8 + 30 µA
	20 Hz - 47 Hz	0.6 + 30 µA	0.9 + 30 µA	1.0 + 30 µA
	47 Hz - 1 kHz	0.07 + 10 µA	0.15 + 20 µA	0.16 + 30 µA
	1 kHz - 10 kHz	0.21 + 30 µA	0.3 + 40 µA	0.4 + 40 µA
330 mA	10 Hz - 20 Hz	1.8 + 400 µA	2.7 + 400 µA	2.8 + 400 µA
	20 Hz - 47 Hz	0.6 + 400 µA	0.9 + 400 µA	1.0 + 400 µA
	47 Hz - 1 kHz	0.1 + 100 µA	0.17 + 180 µA	0.22 + 220 µA
	1 kHz - 10 kHz	0.3 + 300 µA	0.4 + 350 µA	0.6 + 400 µA
2.5 A	10 Hz - 20 Hz	1.8 + 4 mA	2.5 + 4.5 mA	2.7 + 5 mA
	20 Hz - 47 Hz	0.66 + 4 mA	0.8 + 6 mA	0.9 + 6 mA
	47 Hz - 1 kHz	0.6 + 3.8mA	0.63 + 3.8 mA	0.65 + 4 mA
	1 kHz - 10 kHz	0.6 + 4mA	0.62 + 4.5 mA	0.7 + 5 mA

2.5 Resistance Measurements

2.5.1 2-wire and 4-wire

Accuracy ± (% of reading + Ω) [1]

Range [2]	Full Scale 6 ½ Digits	Resolution	Source current	24 hours 23°C ± 1°C	90 Days 23°C ± 10°C	One Year 23°C ± 10°C
33 Ω [3]	33.00000 Ω	10 μΩ	10 mA	0.0038 + 1 mΩ	0.005 + 1.5 mΩ	0.008 + 2 mΩ
330 Ω	330.0000 Ω	100 μΩ	1 mA	0.0037 + 4.5 mΩ	0.0046 + 5 mΩ	0.007 + 6 mΩ
3.3 kΩ	3.300000 kΩ	1 mΩ	1 mA	0.0023 + 28 mΩ	0.003 + 32 mΩ	0.005 + 33 mΩ
33 kΩ	33.00000 kΩ	10 mΩ	100 μA	0.0025 + 300 mΩ	0.0033 + 330 mΩ	0.006 + 350 mΩ
330 kΩ	330.0000 kΩ	100 mΩ	10 μA	0.0055 + 3.2 Ω	0.007 + 4 Ω	0.009 + 5 Ω
3.3 MΩ	3.300000 MΩ	1 Ω	1 μA	0.036 + 70 Ω	0.04 + 75 Ω	0.05 + 80 Ω
33 MΩ	33.0000 MΩ	100 Ω	100 nA	0.22 + 600 Ω	0.24 + 900 Ω	0.26 + 1 kΩ
330 MΩ [3]	330.00 MΩ	10 kΩ	10 nA	1.6 + 50 kΩ	1.8 + 60 kΩ	2.0 + 80 kΩ

[1] With reading rate set to 2 rps or slower, and within one hour of Ohms zero, using Relative control. [2] 4-wire ohms is available up to the 330 kΩ range.

[3] 33 Ω and 330 MΩ ranges are only available with the and MODEL 1005.

2.5.2 6-wire Guarded Resistance Measurement (MODEL 1005)

Typical additional error contributed by guarding

Accuracy ± (% of reading + Ω)

Range	Source current	One Year 23°C ± 5°C [1]
33 Ω	10 mA	0.3 + 4 mΩ
330 Ω	1 mA	0.003 + 20 mΩ
3.3 kΩ	1 mA	0.005 + 100 mΩ
33 kΩ	100 μA	0.03 + 1 Ω
330 kΩ	10 μA	0.35 + 10 Ω

[1] This table should be used in conjunction with the 2-wire and 4-wire table above.

2.6 Leakage Measurement (MODEL 1005)

Accuracy ± (% of reading + Ω) [1]

Leakage Reading	Source Voltage range	One Year 23°C ± 5°C [1]
1.00 ηA to 100.00 ηA	-10 V to +10 V	2 + 350 pA
100.00 ηA to 1000.00ηA	-9 V to + 9 V	1.2 + 2 ηA
1000.00 ηA to 3.3 μA	-7 V to + 7 V	1.5 + 20 ηA

[1] Error does not include external shunt resistor's tolerance.

2.7 RTD Temperature Measurement (MODEL 1005)

RTD Type	Ro (Ω)	Resolution	Temperature range	Temperature Accuracy 23°C ± 5°C [1] One Year
pt385, pt3911, pt3916, pt3926	100, 200 Ω	0.01°C	-150 to 650°C	±0.06°C
pt385, pt3911, pt3916, pt3926	500, 1 kΩ	0.01°C	-150 to 650°C	±0.03°C
Cu (Copper)	Less than 12 Ω	0.01°C	-100 to 200°C	±0.18°C for temperatures ≤ 20°C, ±0.05°C otherwise
Cu (Copper)	Higher than 90 Ω	0.01°C	-100 to 200°C	±0.10°C for temperatures ≤ 20°C, ±0.05°C otherwise

[1] With reading rate set to 2 rps or slower, using a 4-wire RTD. Measurement accuracy does not include RTD probe error.

2.8 Additional Component Measurement Capability

2.8.1 Diode Characterization

Available DC current values 100 ηA, 1 μA, 10 μA, 100 μA and 1 mA. MODEL 1005: 10 mA constant current plus variable current from 10 ηA to 12.5 mA

Typical Current Value Uncertainty 1%

Typical Voltage Value Uncertainty 0.02%

Maximum diode voltage compliance 4 V

2.8.2 Capacitance Measurement (MODEL 1005)

Accuracy ± (% of reading + Farads) [1]

Range	Full Scale 4 ½ Digits	Resolution	One Year 23°C ± 5°C
10 ηF	11.999 ηF	1 pF	2.1 ± 5 pF
100 ηF	119.99 ηF	10 pF	1.0
1 μF	1.1999 μF	100 pF	1.0
10 μF	11.999 μF	1 ηF	1.0
100 μF	119.99 μF	10 ηF	1.0
1 mF	1.1999 mF	100 ηF	1.2
10 mF	11.999 mF	1 μF	2

[1] Within one hour of zero, using Relative control. Accuracy is specified for values higher than 5% of the selected range with the exception of the 10 ηF range, which measures down to 0 pF.

2.8.3 Inductance Measurement (MODEL 1005)

± (% of reading + inductance) [1]

Range	Default frequency	Full Scale 4 ½ Digits	Resolution	Accuracy 23°C ± 5°C One Year [2]
33 μH	75 kHz	33.000 μH	1 ηH	3.0% + 500 ηH
330 μH	50 kHz	330.00 μH	10 ηH	2.0% + 3 μH
3.3 mH	4 kHz	3.3000 mH	100 ηH	1.5% + 25 μH

33 mH	1.5 kHz	33.000 mH	1 µH	1.5% + 200 µH
330 mH	1 kHz	330.00 mH	10 µH	2.5 + 3 mH
3.3 H	100 Hz	3.3000 H	100 µH	3 + 35 mH

[1] Within one hour of zero, and Open Terminal Calibration.

[2] Accuracy is specified for values greater than 5% of the selected range.

2.9 Timing Measurements (MODEL 1005)

2.9.1 Threshold DAC

The Threshold DAC is used for selecting a detection threshold to give optimal frequency and timing measurements.

± (% of setting + volts)

Selected VAC range [1]	Threshold range (DC level)	Threshold DAC resolution	Highest allowed input Vp-p	Typical one year setting uncertainty
330 mV	-1.0 V to +1.0 V	0.5 mV	1.900 V	0.2% + 4 mV
3.3 V	-10.0 V to +10.0 V	5.0 mV	19.00 V	0.2% + 40 mV
33 V	-100.0 V to 100.0 V	50 mV	190.0 V	0.2% + 0.4 V
250 V	-500 V to 500 V	500 V	850.0 V	0.2% + 4 V

[1] This table should be used in conjunction with the AC volts section above.

2.9.2 Frequency and Period Measurement

ACV Mode

Input Impedance 1 MΩ with < 300 pF

Frequency Range	1 Hz - 100 Hz	100 Hz-1 kHz	1 kHz-10 kHz	10 kHz-100 kHz	100 kHz-300 kHz
Resolution	1 mHz	10 mHz	100 mHz	1 Hz	1 Hz
Uncertainty is ±0.002% of reading ± adder shown	4 mHz	20 mHz	200 mHz	2 Hz	5 Hz
Input Signal Range [1]	10% - 200% of range	10% - 200% of range	10% -200% of range	10% - 200% of range	45% -200% of range

[1] Input RMS voltage required for a valid reading. Do not exceed 250 V RMS input. For example, 10% -200% of range indicates that in the 330 mVAC range, the input voltage should be 33 mV to 660 mV RMS.

ACI Mode

Input Impedance 10 Ω in the 3 mA and 30 mA ranges, 0.1 Ω in the 330 mA and 2.5 A ranges.

Frequency Range	1 Hz - 100 Hz	100 Hz-1 kHz	1 kHz-10 kHz	10 kHz-500 kHz
Resolution	1 mHz	10 mHz	100 mHz	1 Hz
Uncertainty	0.01% ±4 mHz	0.01% ±20 mHz	0.01% ±200 mHz	0.01% ±2 Hz
Input Signal Range, 3.3 mA, 330mA Ranges [1]	10% -500% of range	10% - 500% of range	10% -500% of range	10% - 500% of range
Input Signal Range, 33 mA, 2.5A ranges	50% -100% of range	50% - 100% of range	50% - 100% of range	50% - 100% of range

[1] Input current required to give a valid reading. For example, 10% -500% of range indicates that in the 3.3 mA range, the input current should be 0.33 mA to 16.5 mA.

2.9.3 Duty Cycle Measurement

Frequency Range	1 Hz to 100 Hz	100 Hz to 1 kHz	1 kHz to 10 kHz	10 kHz to 100 kHz
Resolution	0.02%	0.2%	2%	20%
Typical Uncertainty is ±0.03% of reading ± adder shown	0.03%	0.3%	3%	20%
Full scale reading	100.00 %	100.00 %	100.00 %	100.00 %

2.9.4 Pulse Width

± (% of reading + sec)

Polarity	Frequency range	Resolution	Width range	Typical Uncertainty
Positive or negative pulse widths	1 Hz to 100 kHz	2 µs	2 µs to 1 s	0.01 +/- 4 µs

2.9.5 Totalizer

Active edge polarity: Positive or negative transition

Maximum count: 10^9

Allowed rate: 1 to 30,000 events per second

Uses Threshold DAC

2.10 Trigger Functions

2.10.1 External Hardware Trigger

Trigger Input voltage level range	+3 V to +15 V
Minimum trigger input current	1 mA
Timing Characteristics	Trigger occurs within 2/Reading rate
Internal Reading Buffer	up to 1,000 readings/sec into 64 readings buffer
Isolation of trigger input	±50 V from analog DMM inputs, and from computer chassis earth ground.

2.10.2 Analog Threshold Trigger

Captures up to 64 readings

Reading rate: 10 rps or higher

2.11 Source Functions (MODEL 1005)

Isolated to 300 V DC from PC Chassis

Current can be paralleled with multiple MODEL 1005s

Voltage can be put in series with multiple MODEL 1005s

2.11.1 DC Voltage Source

Parameter	Closed Loop [1]	Open Loop
Output Voltage range	-10.000 V to +10.000 V	
Typical Current source/sink at 5V output	5 mA	5 mA
DAC resolution	18 bits	12 bits
Accuracy 23°C ± 10°C One Year	0.015% ± 350 µV	1.0% ± 35 mV
Typical settling time	3 S (rate set to 2/s)	1 mS
Typical source resistance	250 Ω	

[1] 10 rps or lower measurement rate is required for the closed loop mode.

2.11.2 AC Voltage Source

Parameter	Closed Loop [1]	Open Loop
Output Voltage, sine wave	50mV to 7.1 V RMS (0.14 to 20.0V peak-to-peak)	
DAC resolution	16 bits	12 bits
Typical Current Drive at 3.5V RMS	3.5 mA RMS	
Accuracy 18°C to 28°C One Year	ACV spec ± 2 mV	ACV spec + 0.8% ± 20 mV
Typical settling time (f-out > 40 Hz)	10 s (rate set to 2 rps)	1.5 s
Typical source resistance	250 Ω	
Frequency range / resolution	2 Hz to 75 kHz / 2 Hz	
Frequency stability	100 ppm ± 1 Hz	

[1] 5 rps or lower measurement rate is required for the closed loop mode.

2.11.3 DC Current Source

Range	Compliance Voltage	Resolution [1]	Minimum level	Accuracy 23°C ± 10°C One Year
1.25 µA	4.2 V	500 pA	1 ηA	1% + 10 ηA
12.5 µA	4.2 V	5 ηA	10 ηA	1% + 100 ηA
125 µA	4.2 V	50 ηA	100 ηA	1% + 500 ηA

1.25 mA	4.2 V	500 ηA	1 µA	1% + 5 µA
12.5 mA	1.5 V	5 µA	10 µA	1% + 50 µA

[1] Resolution without Trim DAC. The use of the Trim DAC can improve the resolution by a factor of 10, but it has to be set separately since it is not calibrated.

2.12 Accuracy Notes

Important All accuracy specifications for DCV, Resistance, DCI, ACV, and ACI apply for the time periods shown in the respective specification tables. To meet these specifications, the System Calibration function must be performed once a day. System Calibration is a simple software operation that takes a few seconds. It can be performed by calling Windows command DMMCal(), or selecting S-Cal in the control panel.

All three products are capable of continuous measurement as well as data transfer rates of up to 1,000 readings per second (rps). To achieve the 6-1/2 digit resolution, the DMM should be operated at 5 rps or slower. The maximum reading rate for 5-1/2 digits is 30 rps.

Accuracy vs. Reading Rates All of the above specifications apply to reading rates of 2 rps or lower. For higher reading rates, increase the noise floor for DCV, Resistance, and DCI by the square root of the increase in reading rate from 2 rps. For example, the noise floor for the 3.3 VDC range is 8 μV at 5 rps. At 20 readings per second, or 10x the reading rate, the noise increases by the square root of 10, or 3.16 times. The noise, then, at 20 readings per second is ± 25 μV.

The noise characteristics for the AC functions increases by the same number as the DC functions. For example, the noise floor for the 3.3 VAC, 20 rps, will have digit rattle of 8.7 mV vs. 2.75 mV at 2 rps.

Reading Rates vs. Noise Rejection The best AC (50 Hz, 60 Hz or 400 Hz) power line rejection is obtained at reading rates that are whole number divisions greater than 1 of the line frequency, as shown in the following table. For best AC line rejection you should use the reading rates checked. It is important to follow this table. Always use the lowest checked rate that is practical for the application.

Reading Rate (rps)	Power Line frequency		
	50 Hz	60 Hz	400 Hz
0.1	√	√	√
0.2	√	√	√
0.5	√	√	√
1	√	√	√
2	√	√	√
5	√	√	√
10	√	√	√
15		√	
20		√	√
25	√		√
30		√	
40			√
50	√		√

60		√	
80			√
100			√
200			√
400			√

Reading Rates vs. Digits of Resolution For reading rates of 10 readings per second (rps) and slower, the DMM has 6 ½ digits of resolution. For reading rates from 10 rps to 30 rps, the DMM has 5 ½ digits of resolution.

2.13 Other Specifications

Temperature Coefficient, All Functions	Less than 0.1 x accuracy specification per °C at 23C ± 5°C
Reading Rate (user selectable)	• 0.5 to 1,000 readings per second (rps) • Up to 10 rps, 6 ½ digits • Up to 30 rps, 5 ½ digits
Hardware Interface	VXI Bus
Overload Protection (voltage inputs)	300 VDC, 250 VAC
Isolation	300 VDC, 250 VAC from Earth Ground

Maximum Input (Volt x Hertz) $8x10^6$ Volt x Hz normal mode input (across Voltage HI & LO).

$1x10^6$ Volt x Hz Common Mode input (from Voltage HI or LO relative to Earth Ground).

Safety Designed to IEC 1010-1, Installation Category II.

Calibration All calibration constants are stored in a text file.

Temperature Range	0°C to 50°C, operating
Size	8.2” X 4.4”
DMM Internal Temperature Measurement (MODEL 1005)	±2°C
Power	+5 volts, 300 mA maximum

Note: Ascor reserves the right to make changes in materials, specifications, product functionality or accessories without notice.

Accessories

Several accessories are available for the MODEL 1004 and the MODEL 1005 DMMs, which can be purchased directly from Ascor. These include:

DMM probes

Deluxe DMM probe set

3.0 Getting Started

After unpacking the 38XX VXI module, please inspect for any shipping damage that may have occurred, and report any claims to your transportation carrier.

The 38XX is shipped with the Digital Multimeter module; USB FLASH drive which contain the various software panels and drivers plus the calibration data specific for the unit, and this Operator's manual.

3.1 Setting the DMM

The MODEL 1004 series DMMs are VXI plug-and-play devices and do not require any switch settings, or any other adjustments to the DMM prior to installation.

The **SM40CAL.DAT** file supplied with your DMM has a unique calibration record for that DMM. (See "**Calibration**" at the end of this manual.) When using multiple DMMs in the same chassis, the **SM40CAL.DAT** file must have a calibration record for each DMM. You must append the unique calibration records of each DMM into one **SM40CAL.DAT** file using a text editor. In general, the **SM40CAL.DAT** file should be placed at the root C:\ directory.

3.2 Installing the Software

It is recommended that you plug the DMM(s) into the VXI chassis, then turn on the computer power. The first time you power up our computer with the DMM installed, your computer will detect the new DMM and prompt you for a driver. the driver your computer requires is located on the USB FLASH drive **(AS3801folder\1_0_4\SETUP.EXE).**

To install the software, run the **'SETUP'** program provided on the USB FLASH drive. This takes care of all installation and registration requirements of the software. If you are installing the DMM on a computer that had a previous version of the software, you should first uninstall the old software. Also make sure you backup and remove the old calibration record (SM40CAL.DAT).

3.3 Installing the 38XX VXI Module

Follow the instructions in the Section 3 of this manual entitled "Installing your VXI Module"

Section 3.0 and 3.2 updated to reflect USB FLASH instead of FLOPPY DISK

3.4 DMM Input Connectors

Before using the DMM, please take a few moments and review this section to understand where the voltage, current, or resistance and other inputs and outputs should be applied. **This section contains important information concerning voltage and current limits. Do not exceed these limits, as personal injury or damage to the instrument, your computer or application may result.**

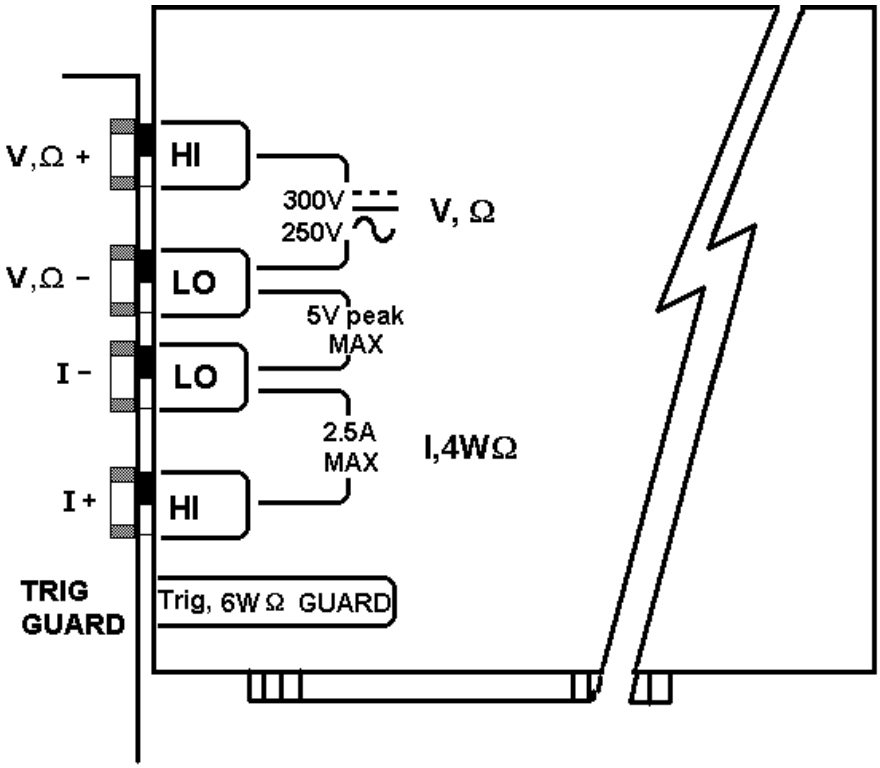


Figure 3-1. The DMM input connectors.

V, Ω + This is the positive terminal for all Volts, 2WΩ, capacitance, diode and inductance measurements, and for sourcing of VDC, VAC and IDC. It is also the Source HI for 4WΩ measurements. The maximum input across **V, Ω +** and **V, Ω -** is 300 VDC or 250 VAC when in the measuring mode**. When in the sourcing mode, the maximum input allowed before damage occurs is 100 volts.**

V, Ω - This is the negative terminal for all Volts, 2WΩ, capacitance diode and inductance measurements, and or sourcing of VDC, VAC and IDC. It is also the Source LO for 4WΩ. **Do not float this terminal or any other DMM terminal more than 300 VDC or 250 VAC above Earth Ground.** (Also, see **Trig, 6W Guard** below.)

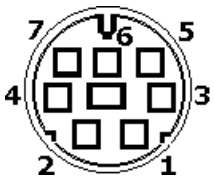
I + This is the positive terminal for all Current measurements. It is also the Sense HI for 4WΩ measurements and 6WΩ guarded measurements. The maximum input across **I, 4WΩ +** and **I, 4WΩ -** is **2.5 A**. Do not apply more than 5 V peak across these two terminals!

I – This is the negative terminal for all Current measurements. In the Current modes, it is protected with a **2.5 A, 250 V Fast Blow fuse** (5 x 20 mm). It is also the Sense LO for 4WΩ measurements and 6WΩ guarded measurements. **V, Ω -** and **I, 4WΩ -** should never have more than 5 V peak across them.

TRIG GUARD Both the Trigger and Guard functions use the DIN-7 connector. This group of pins include the positive and negative hardware trigger input lines and the two MODEL 1005 Guarded Measurement Force and Sense signals. The external trigger initiates reading(s) into the onboard buffer, and the 6W guard signals facilitate in-circuit resistor measurements by means of isolating a loading node. The DIN-7 plug can be ordered from Ascor and is also available at many electronic hardware distributors. The connector is generically referred to as a mini DIN-7 male. The trigger signal should be in the range of 3 V to 12 V peak. The two 6W guard signals should never have more than 5 V peak across them.

Warning! The DIN connector pins are protected to a maximum of 35 V with respect to the PC chassis and any other DMM terminal. Do not apply any voltages greater than 35 V to the DIN connector pins. Violating this limit may result in personal injury and/or permanent damage to the DMM.

DIN-7, Pin number	Function
7	External Trigger, Positive terminal
4	External Trigger, Negative terminal
1	Guard Source (MODEL 1005)
6	Guard Sense (MODEL 1005)



DIN-7 Connector Pin Description, view from bracket side.

3.5 Starting the Soft Front Panel

You can verify the installation and gain familiarity with the DMM by exercising its measurement functions using the VXI *Plug&Play* Soft Front Panel (SFP). To run the SFP, at the Windows Interface click on **START**, then click on **PROGRAMS**, then click on **VXIPNP**, then click on **as3801 Front Panel.**

If you do not hear the relays click, it is most likely due to an installation error. Another possible source for an error is that the **SM40CAL.DAT** file does not correspond to the installed DMM.

The SFP is operated with a mouse. All functions are accessed using the left mouse button. When the DMM is operated at very slow reading rates, you may have to hold down the left mouse button longer than usual for the program to acknowledge the mouse click.

Note: The soft front panel powers up in DCV, 330 V range. If the DMM is operated in Autorange, with an open input, you may hear the DMM relays clicking every few seconds, as a range change occurs. This is perfectly normal with ultra high impedance DMMs. This phenomenon is caused by the virtually infinite input impedance of the 330 mV and 3.3 V DCV ranges. On these ranges, an open input will read whatever charge is associated with the signal conditioning of the DMM. As this electrical charge changes, the DMM will change ranges, causing the relay clicking. This is normal.

3.6 Using the Soft Front Panel

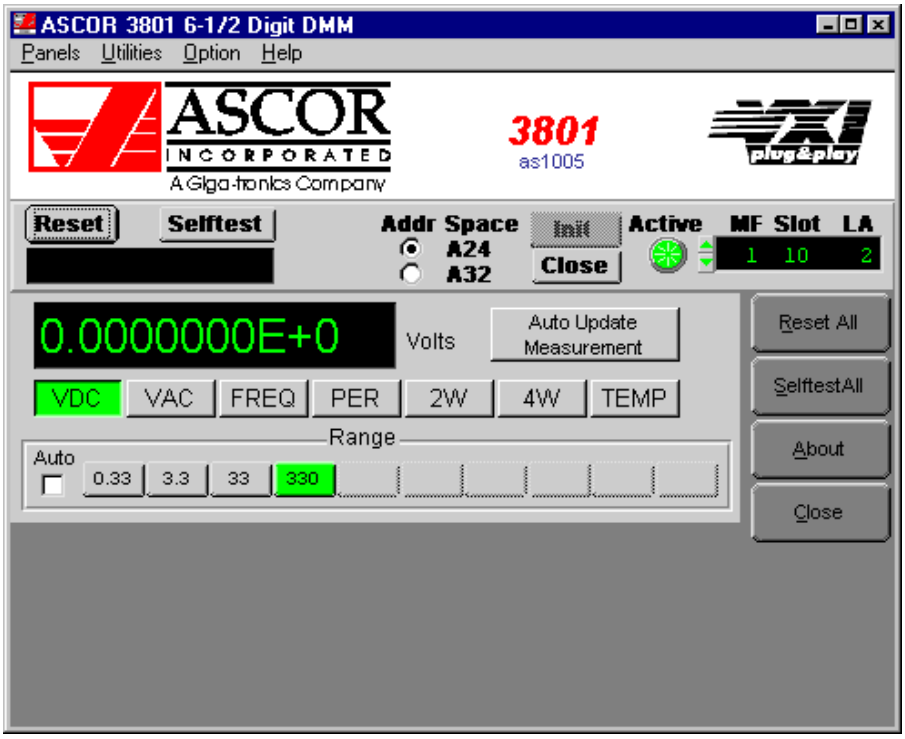


Figure 3-2. The Soft Front Panel for the DMM.

The three main groups include Measure, Source and Range buttons. The 8 Range buttons are context sensitive such that only “330m, 3.3, 33 and 250 appear when in AC Voltage Functions, “3.3m 33m 330m 2.5” appear when in Current Functions, etc.

Note: All of the controls described below correspond to their respective software function, which can be invoked within your control software or as objects in a visual programming environment. The software command language of the DMM provides a powerful set of capabilities. Some of the functions are not included in the control panel, but are in the software.

Source : The source buttons control the type of measurement being performed. The source selections are as follows:

VDC = DC Volts; **VAC** = AC Volts; **FREQ** = Frequency; **PER** = Period; **2W** = Two-Wire Resistance; **4W** = Four-Wire resistance; **TEMP** = Temperature

Range : Can be set to **AutoRange** or manual by clicking on the appropriate range in the lower part of the Windows panel. Autoranging is best used for bench top application and is **not recommended** for an automated test application due to the uncertainty of the DMM range, as well as the extra time for range changes. Locking a range is highly

recommended when operating in an automated test system, especially to speed up measurements. Another reason to lock a range is to control the input impedance in DCV. The 330 mV and 3.3 V ranges have virtually infinite input impedance, while the 33 V and 330 V ranges have 10 MΩ input impedance.

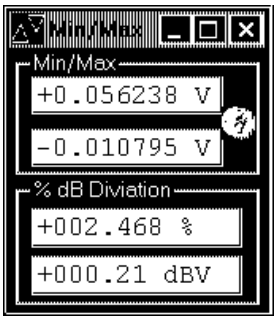
Auto Update Measurement : When this function is **OFF** the DMM retains the last reading.

When this function is ON new readings are taken and displayed approximately ten times a second.

The following functions have not been implemented in the Soft Front Panel as of the date and revision of this manual.

Any reference to these functions in subsequent sections should be ignored at this time.

Relative This is the Relative function. When activated, the last reading is stored and subtracted as reference from all subsequent readings. This is a very important function when making low level DCV measurements, or in 2WΩ. For example, when using 2WΩ, you can null out lead resistance by shorting the leads together and clicking on **Relative.** When making low level DC voltage measurements (e.g., in the μV region), first apply a copper short to the **V,Ω +** & **-** input terminals, allow the reading to stabilize for a few seconds, and click on **Relative**. This will correct for any offsets internal to the DMM. The **Relative** button can also be used in the Percent and dB deviation displays (shown below), which are activated using the **Tools** in the top menu.



The Min/Max box can be used to analyze variations in terms of Min, Max, Percent and dBV. This display can be activated by selecting the Min/Max/Deviation from the Tools menue. For instance, testing a circuit bandwidth with an input of 1V RMS, activate the Relative function with the frequency set to 100Hz, than sweep gradually the frequency, and monitor the percent deviation as well as the dBV error and capture any response anomalies with the Min/Max display. The left display indicates peaking of 2.468% (0.21 dBV) and maximum peaking in the response of +56.24mV and a notch of –10.79mV from the reference at 100Hz.

Rate Box Controls the DMM reading rate. 0.1 rps to 1,000 rps can be set. As measurement rate increases, so does the measurement noise. For best accuracy set to the lowest rate acceptable for the application. Also consider the line frequency (50/60 Hz) of operation when setting reading rates, as certain reading rates have more noise rejection at either 50 or 60 Hz. (See "Specifications" for details.) Set the measurement rate as low as practical for the application. When measuring RMS values, there is no point setting the measurement rate to a value higher than 5 rps since the RMS circuitry has a settling time that is over a second. The capacitance function is not affected by rate setting. For inductance measurements use 10 rps or slower measurement for best accuracy.

Note on Measurement Rate: All three products are capable of continuous measurement as well as data transfer rates of up to 1,000 rps. To achieve the 6-1/2 digit resolution and accuracy, the DMM should be operated at 10 rps or slower. The maximum reading rate for 5-1/2 digits is 30 rps.

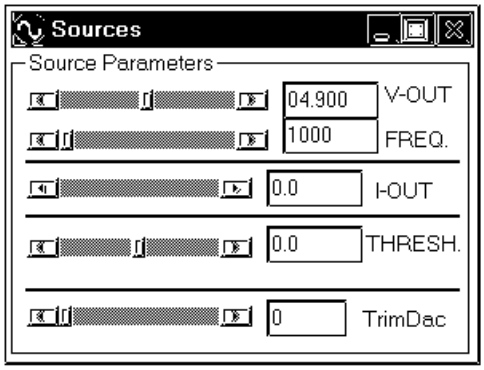
S_Cal This function is the System Calibration that corrects for internal gain, scale factor and zero errors. The DMM does this by alternatively selecting its local DC reference and a zero input. It is required at least once every day to meet the MODEL 1004 accuracy specifications. We recommend that you also perform this function whenever the external environment changes (e.g. the temperature in your work environment changes by more than 5°C, or the , MODEL 1005 on board temperature sensor indicates more than a 5°C change). This function takes less than a few seconds to perform. Disconnect all leads to the DMM before doing this operation. Keep in mind that this is not a substitute for periodic calibration, which must be performed with external standards.

ClosedLoop This check box selection is used in conjunction with the AC and DC Voltage-Source functions of the MODEL 1005. When checked, the DMM monitors the output level and continuously applies corrections to the output level. When not checked, the DMM is a 12-bit source vs. 16 bits in the ClosedLoop mode.

OpenCal This check box selection is used in conjunction with inductance measurement. It is necessary to perform Open Terminal Calibration using this control, prior to measuring inductance. This function characterizes both the internal DMM circuitry as well as the probe cables. To perform OpenCal, attach the probe cables to the DMM, leaving the other end of the probe cables open circuited. Then, activate the OpenCal button.

Sync With this check box selection is active, the DMM measurements are internally synchronized, which reduces the measurement rate, but allows full-scale input swings to be settled in single measurement.

Sources Panel There are three function buttons in the Source group (MODEL 1005 only). The **V, I, LEAK** buttons select one of three source functions, Voltage (DC and AC), IDC and Leakage. The **Sources Panel** is automatically enabled when one of the source functions is enabled. It can also be invoked using the **Sources Panel** selection under the **Tools** menu. This panel allows the entry of values for all of the source functions, including Leakage.



The V-OUT **Scroll bar and Text box are used to set the Voltage for DC and AC Volts as well as for Leakage. When sourcing ACV, the voltage is in RMS and the** FREQ. **Scroll bar and Text box control the frequency of the source. It is also used to control inductance frequency. When sourcing DC current, use the** I-OUT **set of controls. When measuring timing or freqeuncy the** THRESH **set of controls is used for comperator threshold. All of the source controls are context sensitive and will be enabled when appropriate.**

4.0 DMM Operation and Measurement Tutorial

Most of the DMM measurement functions are accessible from the Soft Front Panel (Figure above). All of the functions are included in the Windows DLL driver library. To gain familiarity with the DMM, run the Windows 'SETUP.EXE' to install the software, then run the DMM, as described in the previous section. This section describes in detail the DMM's operation and measurement practices for best performance.

4.1 Voltage Measurement

Measures from 0.1 μV to 300 VDC or 250 VAC. Use the **V, Ω +** and **V, Ω -** terminals, being certain to always leave the **I+, I-** and DIN-7 terminals disconnected. Use the AC/DC button on the Control Panel to switch between AC and DC.

Making Voltage Measurements is straightforward. The following tips will allow you to make the most accurate voltage measurements.

4.1.1 DC Voltage Measurements

When making very low level DCV measurements (<100 μV), you should first short the DMM with a copper wire shorting plug across the **V, Ω +** and **V, Ω -** terminals and perform the **Relative** function to eliminate zero errors before making your measurements. A common source of error can come from your test leads, which can introduce several μVolts of error due to thermal voltages. To minimize thermal voltaic effects after handling the test leads; you should wait a few seconds before making measurements. Ascor offers several high quality probes that are optimal for low level measurements.

Note: The DMM front panel powers up in DCV , 330 V range. If the DMM is operated in Autorange, with an open input, you may hear the DMM relays clicking every few seconds, as a range change occurs. This is perfectly normal with ultra high impedance DMMs. This phenomenon is caused by the virtually infinite input impedance of the 330 mV and 3.3 V DCV ranges. On these ranges, an open input will read whatever charge is associated with the signal conditioning of the DMM. As this electrical charge changes, the DMM will change ranges, causing the relays to click. This is normal.

4.1.2 True RMS AC Voltage Measurements

ACV is specified for signals greater than 1mV, from 10 Hz to 100 kHz. The ACV function is AC coupled, and measures the true RMS value of the waveform. As with virtually all true-RMS measuring meters, the MODEL 1004 may not read a perfect zero with a shorted input. This is normal.

ACV measurements, if possible, should have the NEUTRAL or GROUND attached to the MODEL 1004 **V,Ω -** terminal. See Figure 4-1, below. This prevents any "Common Mode" problems from occurring (Common Mode refers to floating the DMM **V,Ω LO** above Earth Ground.) Common Mode problems can result in noisy readings,

or even cause the PC to hang-up under high V x Hz input conditions. In many systems, grounding the source to be measured at Earth Ground (being certain to avoid any ground loops) can give better results.

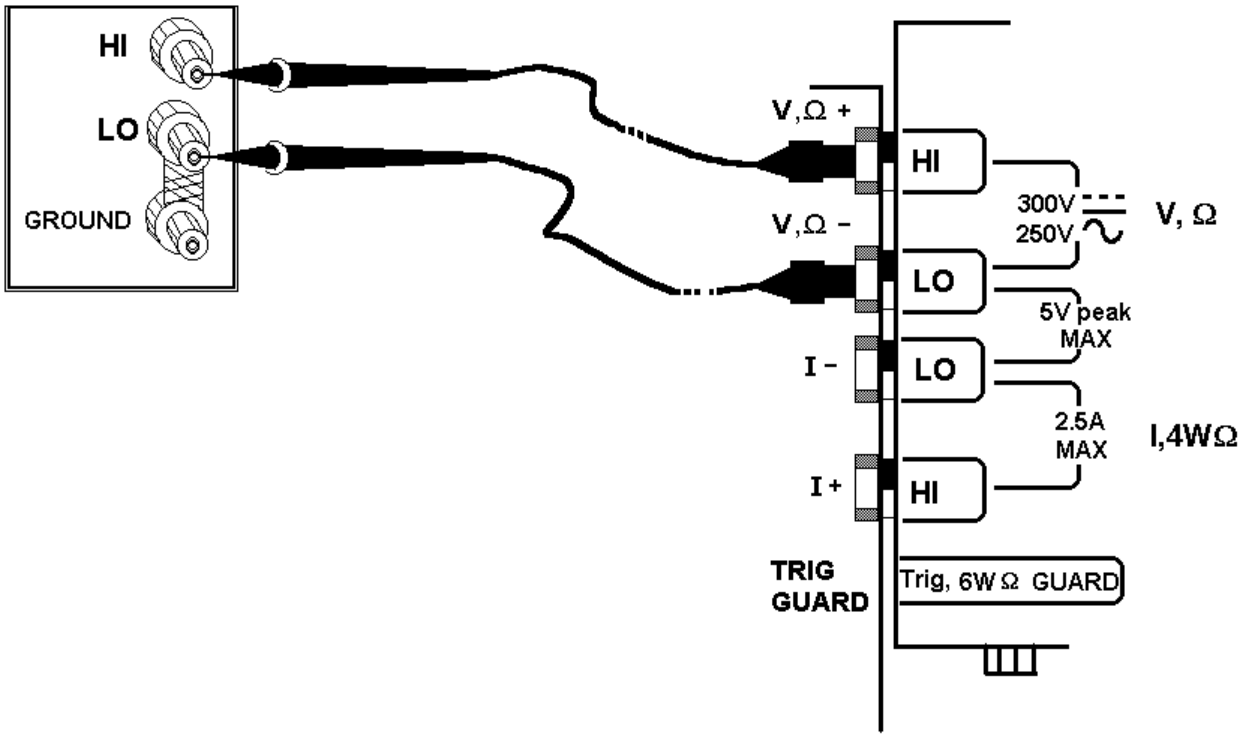


Figure 4-1. Make Voltage ACV measurements with the source ground attached to the MODEL 1004 **V,Ω -** to minimize "Common Mode" measurement problems.

4.1.3 AC Peak-to-Peak and Crest Factor Measurement *(MODEL 1005)*

Measurement of Peak-to-Peak, Crest Factor and AC Median values requires a repetitive waveform between 30 Hz and 100 kHz. The DMM must be in AC voltage measurement mode, with the appropriate range selected. Knowing the Peak-to-Peak value of the waveform is useful for setting the Threshold DAC (described below). This latter function is a composite function, and may take over 10 seconds to perform.

4.1.4 AC Median Value Measurement (MODEL 1005)

To better understand the usage of this function, you should note that the DMM makes all AC voltage measurements through an internal DC blocking capacitor. The voltage is thus "AC coupled" to the DMM. The measurement of the Median value of the AC voltage is a DC measurement performed on the AC coupled input signal. This measurement returns the mid-point between the positive and negative peak of the waveform. The Median value is used for setting the comparator threshold level for best counter sensitivity and noise immunity. (It is difficult to measure the frequency of a low duty cycle, low amplitude AC signal since there is DC shift at the comparator input due to the internal AC coupling. The and MODEL 1005 overcome this problem by allowing you to set the

comparator threshold level). For further information on the usage of AC Median value and Peak-to-Peak measurements, and the Threshold DAC, see the “Frequency and Timing Measurements” section below.

This function requires a repetitive signal. The DMM must be in AC voltage measurement mode, with the appropriate range selected.

4.2 Current Measurements

The MODEL 1004 measures AC and DC currents between 100 ηA and 2.5 A. Use the **I, 4WΩ** terminals, being certain to always leave the **V,Ω +** & **-** terminals disconnected. Use the AC/DC button to switch between AC and DC. The AC current is an AC coupled True RMS measurement function.

The Current functions are protected with a 2.5 A, 250 V fuse. The 3.3mA and 33mA ranges utilize a 10Ω shunt, while the 330mA and 2.5A ranges use a 0.1Ω shunt. In addition to the shunt resistors, there is some additional parasitic resistance in the current measurement path associated with the fuse and the internal wiring.

Warning! Applying voltages > 35 V to the I+, I- inputs can cause personal injury and/or damage to your DMM and computer! Think before applying any inputs to these terminals!

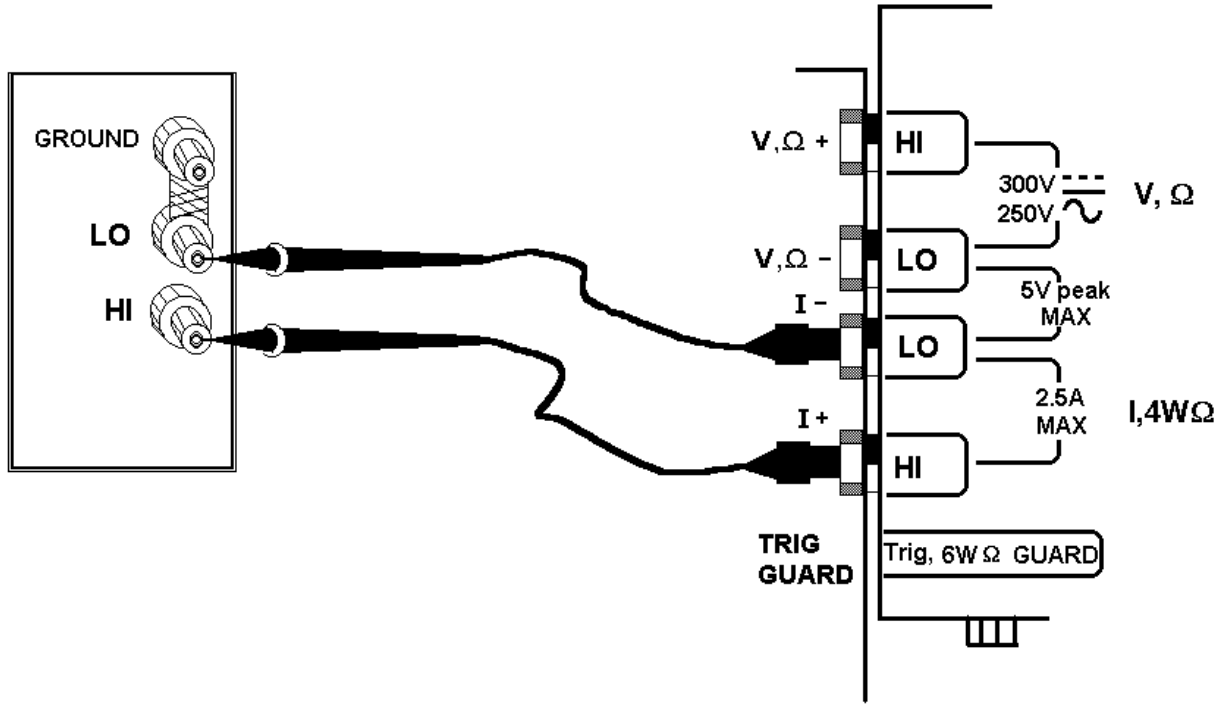


Figure 4-2. AC and DC Current measurement connection.

4.2.1 Improving Current Measurements

When making sensitive DC current measurements, be sure to use the **Relative** function to zero out any residual errors of the MODEL 1004. This is easily accomplished by disconnecting all terminals to the DMM and performing **Relative** in the appropriate DCI range. Using the **S-Cal** (DMMCalibrate()) prior to **Relative** will improve accuracy further. Although the MODEL 1004 family is designed to withstand up-to 2.5A indefinitely, be aware that excessive

heat may be generated when measuring higher AC or DC currents. If allowed to rise this heat may adversely effect subsequent measurements. In consideration with this effect, it is recommended that whenever practical, higher current measurements be limited to short time. The lower two ranges of DC current are effected by relay contamination. If the measurements seem high, apply between 2mA and 5mA to the current terminals and alternate between the 3.3mA and 330mA ranges. Repeat this for at least 100 times. Best do this under program control with measurement rate set to 200rps. This will clean the relay contacts from oxides and contaminants.

4.2.2 Low Level DC Current Measurements

For low level current measurements use the **V, Ω+** and **V, Ω-** terminals. Using the 33V DCV range, the MODEL 1004 can measure very low currents. This hidden measurement function is facilitated by the DMM's low leakage front-end and a virtual 10.0MΩ input resistance. With a typical offset error of less than 100μV in this VDC range, it is practical to measure down to 20pA. The maximum current value that can be measured has more to do with the user's acceptable burden voltage (the voltage drop across the 10.0MΩ shunt) then the DMM limitations. Assuming a maximum burden voltage of 3.3V the maximum current level is 330ηA. This rage is well within leakage measurements required in semiconductor testing. It is also a very quite and stable. Since the DMM does not have an explicit low current function, it is necessary to calculate the current. It is equal to the measured voltage divided by 10.0MΩ.

4.3 Resistance Measurements

Resistance is measured with one of seven (six in the MODEL 1004) precision current sources, with the DMM displaying the resistance value. Most measurements can be made in the 2-wire mode. The 4-wire ohms is used to make low value resistance measurements.

4.3.1 2-wire Ohm Measurements

Measures from 100 μΩ to 33 MΩ (10 μΩ to 330 MΩ in the , MODEL 1005). Use the **V,Ω+, V,Ω-** terminals, being certain to always disconnect the **I+, I-** terminals.

Most resistance measurements can be made using the simple 2-wire Ohms method. Simply connect **V,Ω+** to one end of the resistor, and the **V,Ω-** to the other end. If the resistor to be measured is less than 30 kΩ, you should null out any lead resistance errors by first touching the **V,Ω+** and **V,Ω-** test leads together and then performing a **Relative** function. If making measurements above 300 kΩ, you should use shielded or twisted leads to minimize noise pickup. This is especially true for measurements above 1 MΩ.

You may also want to control the Ohms current used in making resistance measurements. (See the Specifications section, "Resistance, 2-wire and 4-wire", for a table of resistance range vs. current level.) All of the Ohms ranges of the MODEL 1004 have enough current and voltage compliance to turn on diode junctions. For characterizing semiconductor part types, use the Diode measurement function. To avoid turning on a semiconductor junction, you may need to select a higher range (lower current). When checking semiconductor junctions, the DMM displays a resistance value linearly related to the voltage across the junction.

For applications requiring resistance measurements higher than 330 MΩ, the Extended Resistance Measurement method is available with the MODEL 1005.

4.3.2 4-wire Ohm Measurements

4-wire Ohms measurements are advantageous for making measurements below 330 kΩ, eliminating lead resistance errors. The **Voltage** (**V,Ω**) Input terminals serve as the current "Source" (i.e. they provide the current stimulus in the ohms measurement), and the **I, 4WΩ** Input terminals are the "Sense" inputs. The Source + and Sense + leads are connected to one side of the resistor, and the Source - and Sense - leads are connected to the other side. Both Sense leads should be closest to the body of the resistor. See Figure 4-3.

4-wire Ohm makes very repeatable low ohms measurements, from 100 μΩ (10 μΩ for MODEL 1005) to 330 kΩ. We do not recommend using **4WΩ** when making measurements above 100 kΩ, although 4-wire ohms is allowed up to 330 kΩ. 4-wire measurements are disabled above 330 kΩ since the extra set of leads can actually *degrade* the accuracy, due to additional leakage and noise paths.

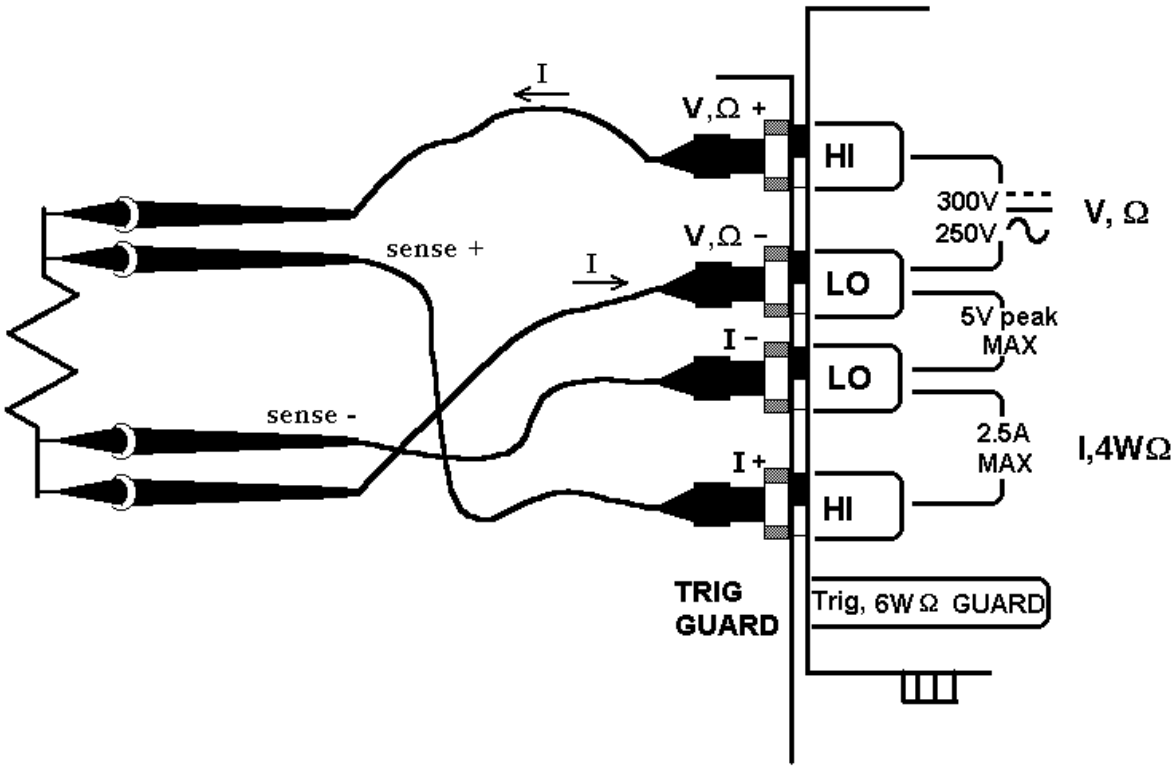


Figure 4-3. The **I- and I+** sense leads should be closest to the body of the resistor when making 4WΩ measurements.

4.3.3 6-wire Guarded Resistance Measurement (MODEL 1005)

The MODEL 1005 provides a guarded 6-wire resistance measurement method. It is used to make resistance measurements when the resistor-under-test has other shunting paths, which can cause inaccurate readings. This method isolates the resistor-under-test by maintaining a guard voltage at a user-defined node. The guard voltage prevents the shunting of the DMM Ohms source current from the resistor-under-test to other components. The Guard Source and Guard Sense terminals are provided at pins 1 and 6 of the DIN connector respectively.

Warning! The DIN connector pins are only protected to a maximum of 35 V with respect to the PC chassis or any other DMM terminal. Do not apply any voltages greater than 35 V to the DIN connector pins. Violating this limit may result in personal injury and/or permanent damage to the DMM.

Example: Assume a 30 kΩ resistor is in parallel with two resistors, a 510 Ω and a 220 Ω, which are connected in series with each other. In a normal resistance measurement, the 510 Ω and 220 Ω would "swamp" the measurement shunting most of the DMM Ohms source current. By sensing the voltage at the top of the 30 kΩ, and then applying this same voltage to the junction of the 510 Ω and 220 Ω, there is no current flow through the shunting path. With this "guarding", the MODEL 1005 accurately measures the 30 kΩ resistor.

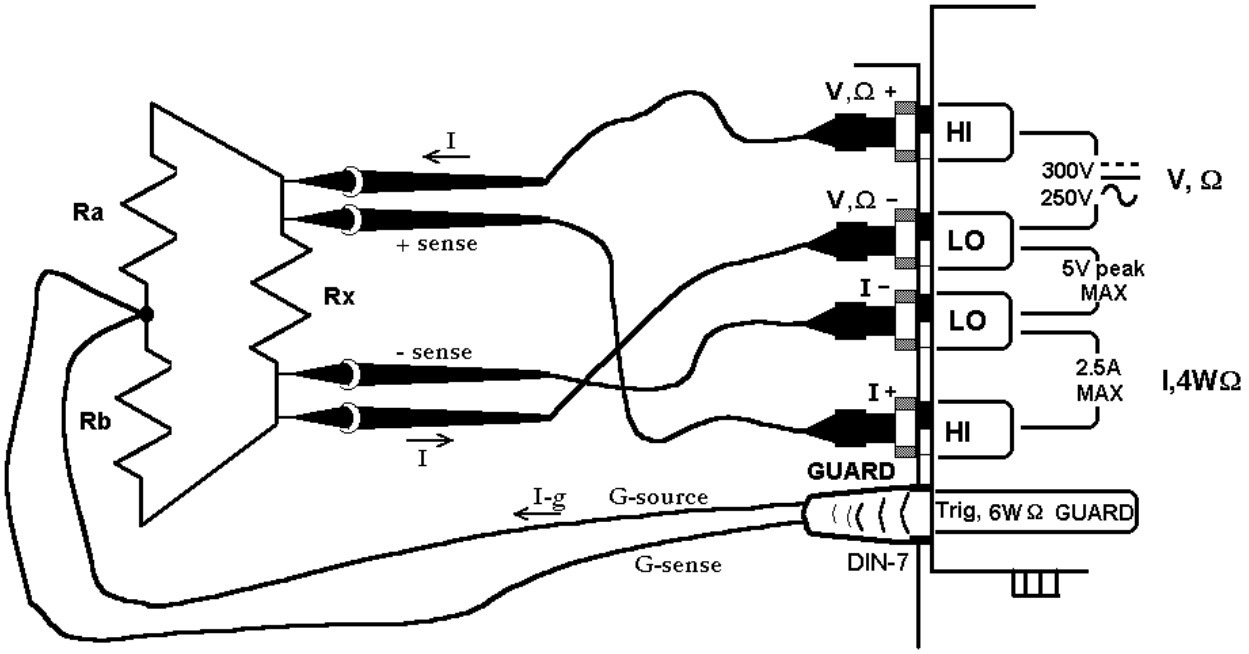


Figure 4-4. 6-wire guarded in-circuit ohms measurement configuration.

The current compliance of the Guard Force is limited to a maximum of 20 mA and is short circuit protected. The resistor connected between the low of the 4-wire terminals and the guard point is the burden resistor, or R_b. Due to the limited guard source current, this resistor can not be lower than R_{bmin}: $R_{bmin} = I_o * R_x / 0.02$, where I_o is the ohms source current for the selected range, and R_x is the resistance being measured. For example, selecting the 330 Ω range and measuring a 300 Ω resistor imposes a limit on R_b of at least 15 Ω or greater. Since the top burden resistor, R_a, does not have this limit imposed on it, selecting the measurement polarity, R_a can become R_b and vise versa. For cases where this limit is a problem, simply set the measurement polarity such that R_a is the higher of the two burden resistors.

To measure values greater than 330 kΩ using the 6-wire guarded method, it is necessary to select the 2-wire ohms function, and maintain the 6-wire connection as in Figure 4-4 above.

4.3.4 Extended Ohms and Leakage Measurements (MODEL 1005)

Leakage Measurement

The MODEL 1005 measures leakage currents by sourcing a DC voltage and measuring current through an external shunt resistor. See Figures 4-5, 4-6 for configuration. Set the DC voltage source using **DMMSetDCVSource()**, and read the leakage current by using **DMMRead()** or **DMMReadNorm()** functions. The leakage voltage maybe set between –10V and +10 V. The leakage currents measurement range is from 1 ηA to 20 mA, depending on the value of the shunt resistor. The value of this shunt resistor has to be conveyed to the DMM by using the **DMMSetLeakageShunt()** function (the default is 1 MΩ). The maximum leakage current is limited to **3.3V/Rshunt**. It is neccessary to repeatedly read the leakage to allow the DMM to make on the fly corrections to the source, compensating for stimulus errors due drift and load variations. Performing open terminal calibration will imrove the accuracy of this function (use DMMOpenTerminalCal() with all terminals open). Refer to Figures 4-5 and 4-6 below. See section 2.6 for specifics.

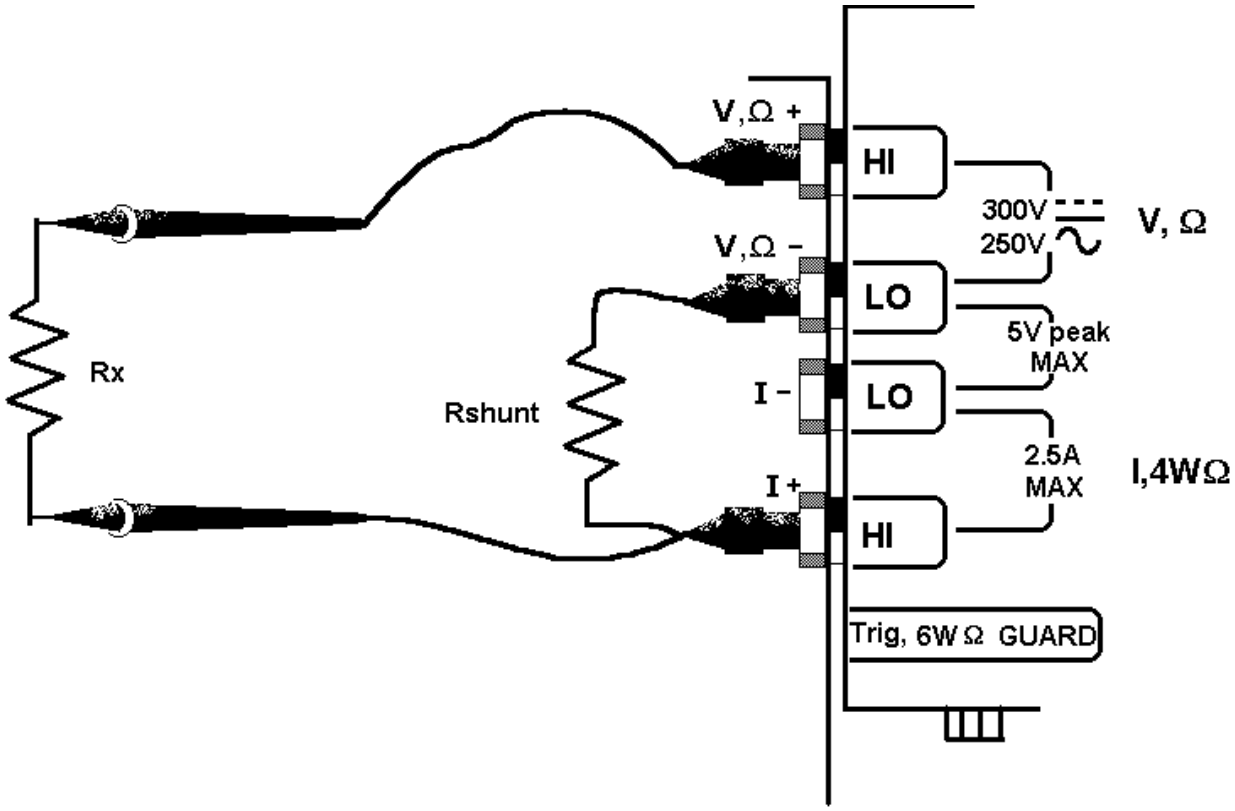


Figure 4-5. Extended Ohms range.

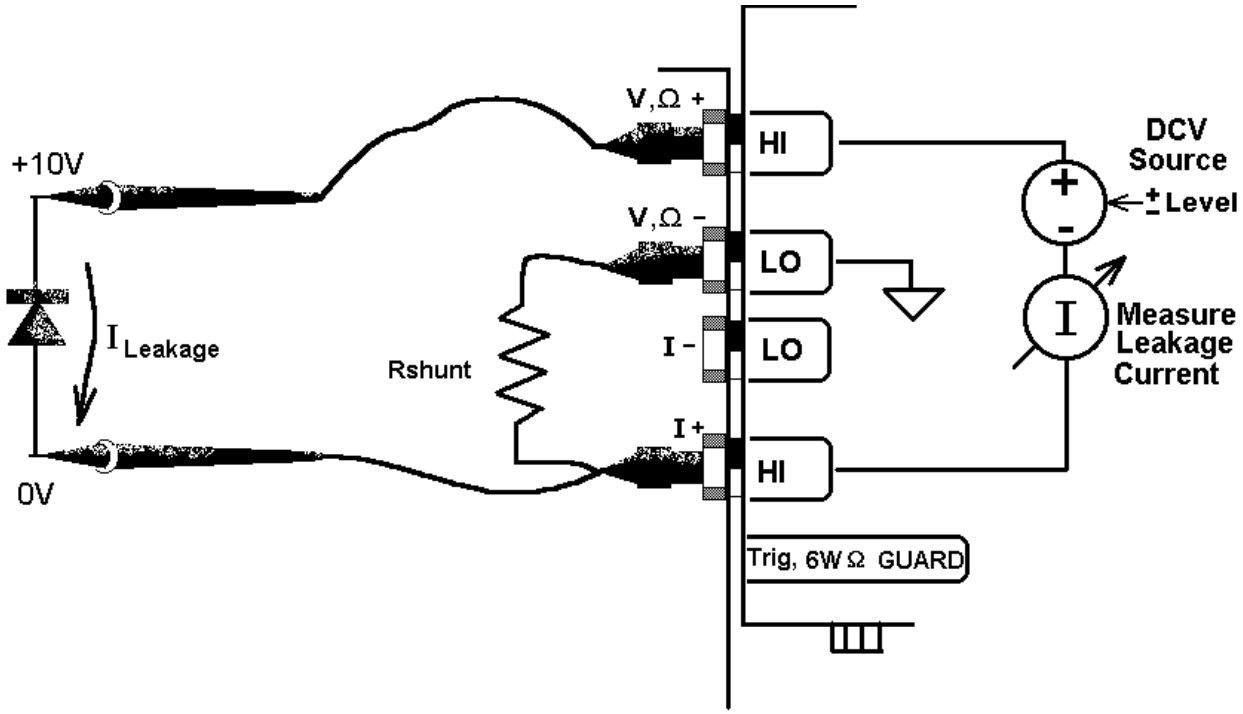


Figure 4-6. Leakage Test Configuration. Measurement of reverse diode leakage at 10V.

Extended Resistance Measurement Methodology

Utilizing the Leakage Measurement configuration in Figure 4-5, high value resistances can be measured. Use the following equation to derive the resistance being measured: $R_x = V_s/I$, where V_s is the DC voltage source, and I is the leakage current being measured. For example, if you set the test voltage to 9.0 V and measure 9 ηA of leakage current, this corresponds to 1,000 MΩ of resistance. This application is useful for testing cables, as well as other leakage sensitive objects such as printed circuit boards**,** connectors and semiconductors. See Leakage measurement function for more details.

4.4 RTD Temperature Measurement (MODEL 1005)

For temperature measurements, the and MODEL 1005 measure and linearize RTDs. 4-wire RTD's can be used by selecting the appropriate RTD type. Any ice temperature resistance between 25 Ω and 10 kΩ can be set for the platinum type RTDs. Copper RTDs can have ice temperature resistance values of 5 Ω to 200 Ω. The highest accuracy is obtained from 4-wire devices, because the resistance of the test leads is nulled out. The connection configuration for RTDs is identical to 4-wire Ohms.

4.5 Internal Temperature (MODEL 1005)

A special on board temperature sensor allows monitoring of the DMM's internal temperature. This provides the means to determine when to run the self-calibration function (S-Cal) for the DMM, as well as predicting the performance of the DMM under different operating conditions. When used properly, this measurement can enhance

the accuracy and stability of the DMM. It also allows monitoring of the PC internal temperature, which is important for checking other instruments in a PC-based test system.

4.6 Diode Characterization

The Diode measurement function is used for characterizing semiconductor part types. This function is designed to display a semiconductor device's forward or reverse voltage. The DMM measures diode voltage at a selected current. The available source currents for diode I/V characterization include five DC current values; 100 ηA, 1 μA, 10 μA, 100 μA and 1 mA. The and MODEL 1005 have an additional 10 mA range. The MODEL 1005 also has a variable current source which can be used concurrently with DCV measurement (see "Source Current / Measure Voltage"). This allows a variable current from 10 ηA to 12.5 mA. The maximum diode voltage compliance is approximately4 V.

Applications include I/V characteristics of Diodes, LEDs, Low voltage Zener diodes, Band Gap devices, as well as IC testing and polarity checking. Typical current level uncertainty for diode measurements is 1%, and typical voltage uncertainty is 0.02%.

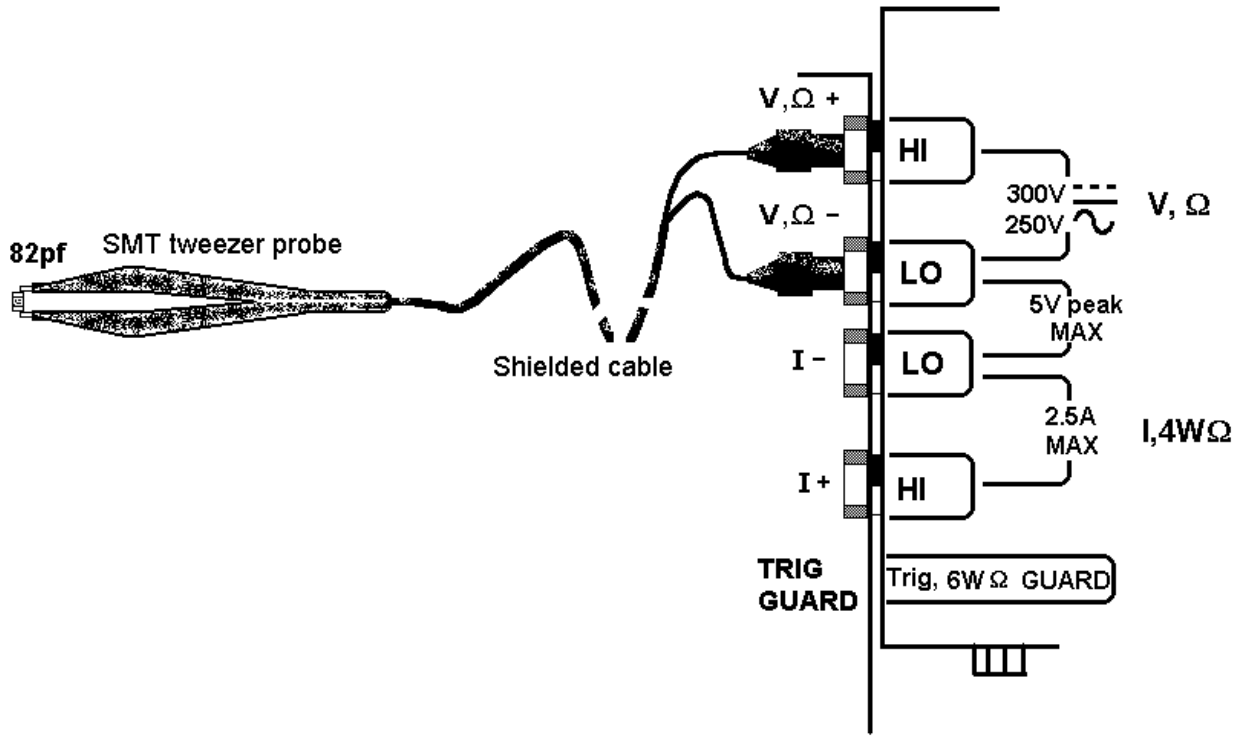


Figure 4-7. Measuring capacitors or inductors is best handled with coaxial or shielded probe wires.

4.7 Capacitance Measurement (MODEL 1005)

The and MODEL 1005 measure capacitance using a differential charge slew method, where variable currents are utilized to produce a dv/dt across the capacitor. Use short high quality shielded probe cables with no more than 500 pF. With the exception of the 10 ηF range, each of the ranges has a reading span from 5% of range to full scale. Capacitance values less than 5% of the selected range indicate zero. Since some large value electrolytic capacitors have significant inductance, as well as leakage and series resistance, the Autoranging function may not be practical.

Because Capacitance measurement is sensitive to noise, you should keep the measurement leads away from noise sources such as computer monitors. For best measurement accuracy at low capacitance values, zero the DMM using the 'Relative' while in the 10 ηF range. The effect of the cable quality and its total capacitance is significant particularly on low value caps. For testing surface mount parts, use the optional Ascor SMT Tweeter probes. You may increase the measurement speed by using the **SetCapsAveSamp()** function.

4.8 Inductance Measurement (MODEL 1005)

The MODEL 1005 measures inductance using a precision AC source with a frequency range of 20 Hz to 75 kHz. Since inductors can vary greatly with frequency, you should choose the appropriate generator frequency. In addition to inductance, the inductor's Q factor can be measured. A high quality coaxial or at least a shielded cable is highly recommended. For best accuracy, perform the Open Terminal Calibration function within an hour of inductance measurements. The Open Terminal Calibration function must be performed with the cables plugged into the DMM, but with the other end open circuited. This process characterizes the internal signal path inside the DMM, the open application cable and the DMM circuitry. Set the measurement rate to 10 rps or lower for best accuracy.

For best measurement accuracy at low inductance values, zero the DMM often by using the '**Relative**' function with the leads shorted. This must be done after Open Terminal Calibration operation. This Relative action measures and removes the inductance of the DMM signal path and that of the application cable.

4.9 Characteristic Impedance Measurement (MODEL 1005)

To measure transmission line's characteristic impedance, measure the cable's capacitance C (with the end of the cable open) and then it's inductance L (with the end of the cable shorted). The cable's impedance equals the square root of L/C. Be certain the cable is long enough such that both the capacitance and inductance are within the specified measurement range of the MODEL 1005.

4.10 Trigger Operation

4.10.1 External Hardware Trigger

The Trigger functions provide for a stand-alone capture of measurements. The local controller supervises the operation, and when conditions are valid, it captures data into its buffer, or sends it back to the PC bus. The reading rate must be set to 10 rps or higher. The External Trigger's isolated high and low input lines are provided at pins 7 (+) and 4 (-), respectively, on the DIN connector. You can abort the External Trigger modes by sending the DMM the Disarm command. The hardware trigger functions include: **DMMArmTrigger**, **DMMSetBuffTrigRead,** and **DMMSetTrigRead**. Read about these functions in the Windows Command Language section (5.6) for details.

Warning! The DIN connector pins are only protected to a maximum of 30 V with respect to the PC chassis or any other DMM terminal. Do not apply any voltages greater than 30 V to the DIN connector pins. Violating this limit may result in personal injury and/or permanent damage to the DMM.

4.10.2 Analog Threshold Trigger

This mode triggers the DMM at a specific input level. A command to the DMM sets a threshold value and arms the DMM Analog trigger. The DMM's local controller waits for the level crossing and captures up to 64 readings, which are saved on board; at the current DMM measurement function, range and rate. The reading rate must be set to 10 rps or higher. You can abort this mode by sending the DMM a Disarm command to the Analog Trigger.

4.10.3 Software Issued Triggered Operations

There are several software trigger functions. They can commend the to make a predefined number of readings, with a specified number of settling readings. These include **DMMSetBuffTrigRead**, **DMMSetTrigRead**, **DMMTrigger**, **DMMBurstRead** and **DMMBurstBuffRead**. Read about these functions in the Windows Command Language section (5.6) for details.

4.11 Frequency and Timing Measurements (MODEL 1005)

While the maximum RMS reading is limited to the set range, you can use most of the timing functions even if the RMS voltage reading indicates overrange. This is true as long as the input peak-to-peak value does not exceed 5.75 times the selected range (5.75 x 330 mV = 1.9 V p-p with the 330 mV range).

4.11.1 Threshold DAC

All timing measurements utilize the AC Voltage path, which is AC coupled. You need to select the appropriate ACV range prior to using the various frequency and timing measurement functions. The and MODEL 1005 have a novel feature to accurately make these measurements for all waveforms. Unlike symmetrical waveforms such as a sine wave and square wave, non-symmetrical waves may produce a non-zero DC average at the frequency counter's comparator input. Other DMMs have the comparator hard-wired to the zero crossing. The and MODEL 1005 include a bipolar, variable Threshold DAC for improved performance of these measurements. The Threshold DAC allows the internal timing comparator to trigger at a specific DC level. Functions affected by the Threshold DAC include frequency, period, pulse-width, duty-cycle and the totalizer.

The Threshold DAC has 12 bits of resolution. Depending on the selected ACV range, this bipolar DAC can be set from a few mV to effectively several hundred volts (referred to the input of the DMM), positive or negative. See the Specifications sections for the limits of AC Median Value measurements and Threshold DAC settings.

The best setting of the Threshold DAC is based on the AC Median Value and Peak-to-Peak measurement described earlier. For example, a 5 V logic signal with 10% duty cycle will result in median value of 2 V, whereas a 90% duty cycle signal will have a –2 V median value. Setting the Threshold DAC to the appropriate median value will result in reliable and accurate timing measurements in each case.

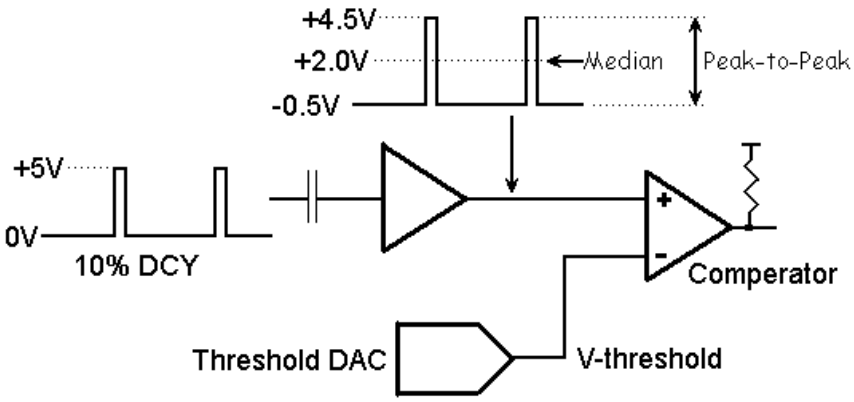


Figure 4-6. AC coupled timing measurements with Threshold DAC.

With the 3.3 ACV range selected, a 10% duty-cycle square wave with 5 V peak-to-peak value, presents a peak-to-peak signal at the internal measuring circuits of –0.5 V to + 4.5 V. The AC Median Value is +2.0 V. By setting the Threshold DAC to the Median value, the internal measuring circuits are properly biased for best performance.

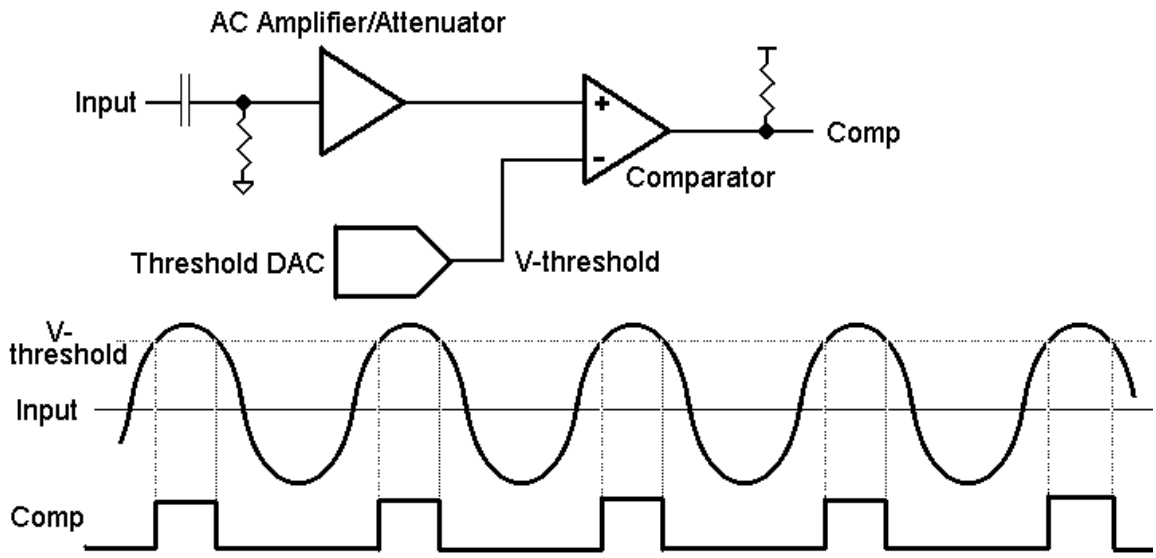


Figure 4-8. Comparator and Threshold DAC Settings

4.11.2 Frequency and Period Measurements

Both **Freq.** and **Per** check boxes are only visible when ACV or ACI functions are selected. These check boxes are used to make frequency or period measurements. **Freq.** measures from 1 Hz to 300 kHz. When activated, the control panel alternately updates the amplitude reading followed by the frequency reading. The reading rate is slower than indicated when frequency is activated. In the Windows control panel, period (**Per**) is also selectable. Once the frequency range is acquired, Frequency and Period have a maximum measurement time of about 1 second. It could take up to five measurements before the correct frequency range is auto-selected. This process is automatic. Once within range, the next frequency measurement is made at the last selected range.

Both Frequency and Period measurement performance can be improved by properly setting the Threshold DAC, a novel feature of the and MODEL 1005. See "Threshold DAC", "AC Median Value", and "Peak-to-Peak" measurements for further details.

4.11.3 Duty Cycle Measurement

Duty Cycle of signals from 1 Hz to 100 kHz can be measured. The minimum positive or negative pulse width of the signal must be at least greater than 2 µs. When measuring duty cycle precisely, the voltage at which the measurement is made is important, due to finite slew rates of the signal. With the and MODEL 1005, the Threshold voltage can be set for precise control of the level at which duty cycle is measured. For best measurement results, set the Threshold DAC to the Median value. This is particularly important for signals with low duty-cycle and small amplitude relative to the selected scale.

4.11.4 Pulse Width

User selectable positive or negative pulse widths may be measured for signal frequencies of 1 Hz to 100 kHz and a minimum pulse widths of 2 μs. The Threshold DAC feature allows measurements at a pre-defined signal level. See Threshold DAC above for more details.

To measure pulse width, the DMM must be in the AC volts range appropriate for the input voltage. Keeping the peak-to-peak amplitude of the measured signal below 5.75 times the set range will guarantee the signal is within the linear region of the AC circuitry and gives the best performance.

4.11.5 Totalizer

The totalizer can be selected while the DMM is in the ACV mode. It is capable of counting events such as over-voltage excursions, switch closures, decaying resonance count, etc. The active edge polarity can be set for a positive or negative transition. A count of up to 10^9 may be accumulated. The maximum rate of accumulation is 30,000 events per second.

The Threshold DAC can be set for a negative or positive voltage value. See Threshold DAC above for more details.

Example One: To monitor and capture the AC line for positive spikes which exceed 10% of the nominal 120 V RMS value, first select ACV 250 V range, than set the Threshold DAC to 186.7 V. This value is the peak value of 120 V RMS plus 10% (120V + 10%) X $\sqrt{2}$). Enable the Totalizer and read it periodically to get the number of times this value was exceeded.

Example Two: Defects in coils, inductors, or transformers can be manifested as an increased decay, or greatly attenuated resonance when stimulated with a charged capacitor. The Totalizer function can be utilized to count transitions above a preset Threshold voltage as in the figure below.

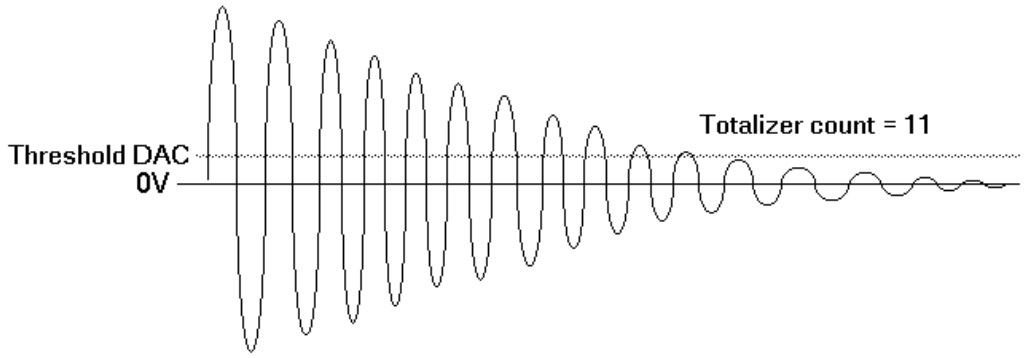


Figure 4-9. Measuring inductor Q by counting transitions of decaying resonance with preset threshold.

4.12 Sourcing Functions (MODEL 1005)

The MODEL 1005 adds a number of sourcing functions, giving greater versatility for a variety of applications. All of the available sources, VDC, VAC, IDC, are isolated (floating with respect to the PC chassis). This allows

sourcing with a significant common mode voltage as well as the ability to connect several MODEL 1005 units in parallel for increased DC current, or in series for increased DC voltage.

Two digital-to-analog converters (DACs) are used for the source functions, a 12 bit DAC, and a Trim DAC. The last augments the 12 bit DAC to form a 16 bit composite DAC and adds an additional 8 bits of resolution. For functions requiring high precision, use both DACs by selecting the ClosedLoop mode, otherwise only the 12 bit DAC is utilized. DCI source is limited to the 12 bit DAC only.

All three source functions use the **V,Ω+**, and the **V,Ω-** terminals of the MODEL 1005.

4.12.1 DC Voltage Source

The MODEL 1005 has a fully isolated bipolar DC voltage source. Two modes of operation are available: fast settling or closed loop. In the ClosedLoop mode the DMM monitors the voltage source output, and updates it using the composite 16 bit DAC, at a rate proportional to the set measurement rate. The ClosedLoop mode offers the best accuracy and resolution. A 10 rps or lower measurement rate is recommended for the ClosedLoop mode. In the fast settling mode, no adjustments are made and the 12 bit DAC is used. Up to ±10.0 V can be sourced, with 10 mA maximum drive. The output source resistance of the DCV source is approximately 250 Ω.

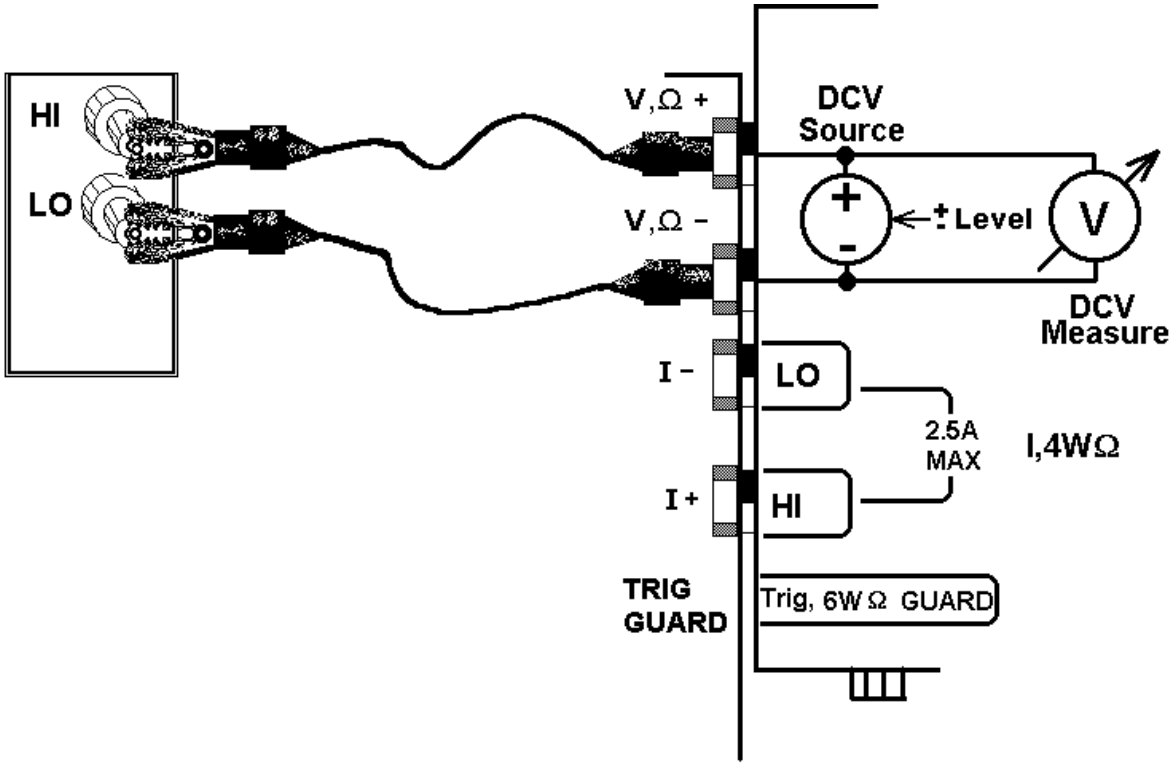


Figure 4-10. Sourcing DC voltage. The figure indicates the internal monitoring of the output in closed loop operation.

4.12.2 AC Voltage Source

The AC voltage source is fully isolated. It has two modes of operation: fast settling or closed loop. In the ClosedLoop mode, the source voltage is monitored, and corrections are made to the composite 16 bit DAC at a rate proportional to the set measurement rate. A 10 rps or lower reading rate is recommended for the ClosedLoop mode. The ClosedLoop mode offers the best accuracy. In the fast settling mode, the source voltage is monitored and can be displayed, but no DAC adjustments are made. Both amplitude and frequency can be set. The frequency range is 2 Hz to 75 kHz, and the amplitude is up to 20 V peak-to-peak with 10 mA maximum peak current drive. The output impedance is approximately 250 Ω.

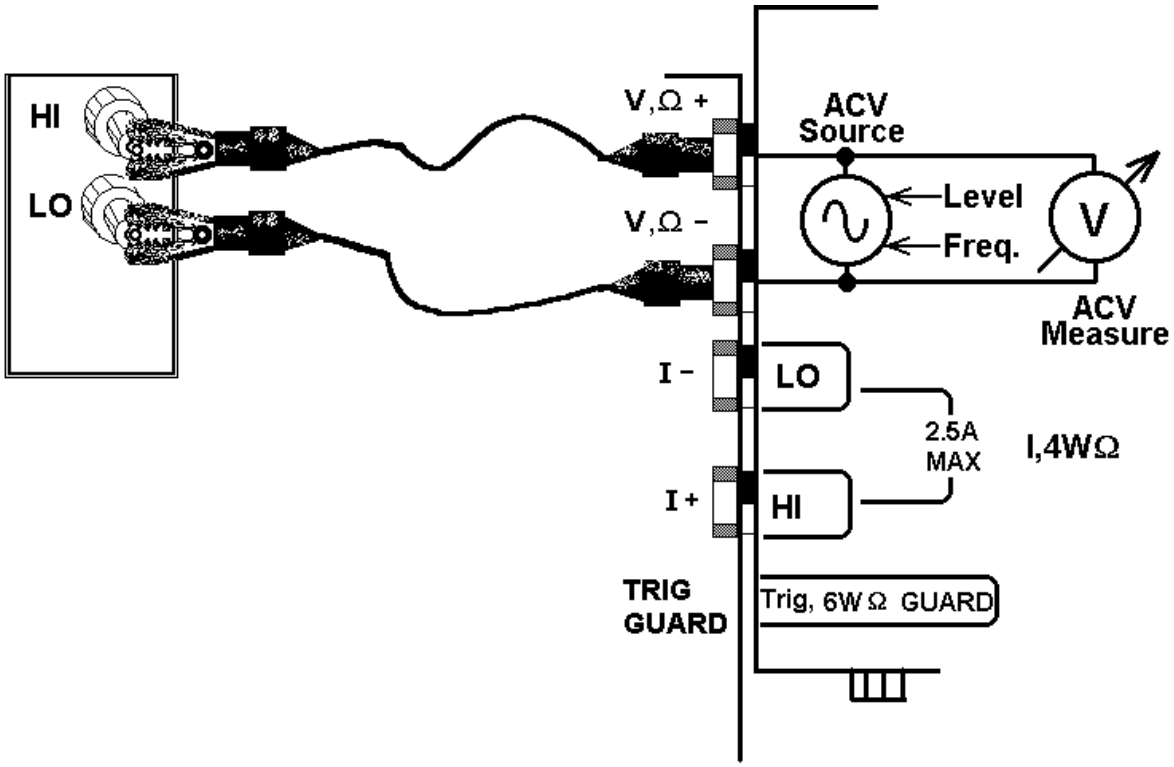


Figure 4-11. Generating AC voltage. The figure indicates the internal monitoring of the output in closed loop operation.

4.12.3 DC Current Source

The MODEL 1005 has a fully isolated unipolar DC current source with five ranges. It uses the 12 bit DAC to control current level. This source function is useful for parametric component measurements as well as for system verification and calibration, where a precise DC current is necessary to calibrate current sensing components.

For improved resolution of the current source, use the Trim DAC. It has to be set separately, since it is not included in the calibration record, or the control software. Use DMMSetTrimDAC() command with a parameter of 0 to 100. Further details are in Chapter 6.

4.12.4 Source Current - Measure Voltage

When sourcing current and measuring voltage, there are two connection configurations: 1) Four wire connection, where the current sourcing terminals and the voltage sense terminals are connected to the load, as in 4-wire Ohms measurement function; and 2) Two wire connection, where the current source terminals also serve as voltage sense probes as in the 2-wire Ohms measurement configuration. The first method eliminates lead resistance errors. One application is in semiconductor diode characterization discussed in Component Testing above. See Current Source Output for range details. Voltage compliance is limited to 4 V in both configurations.

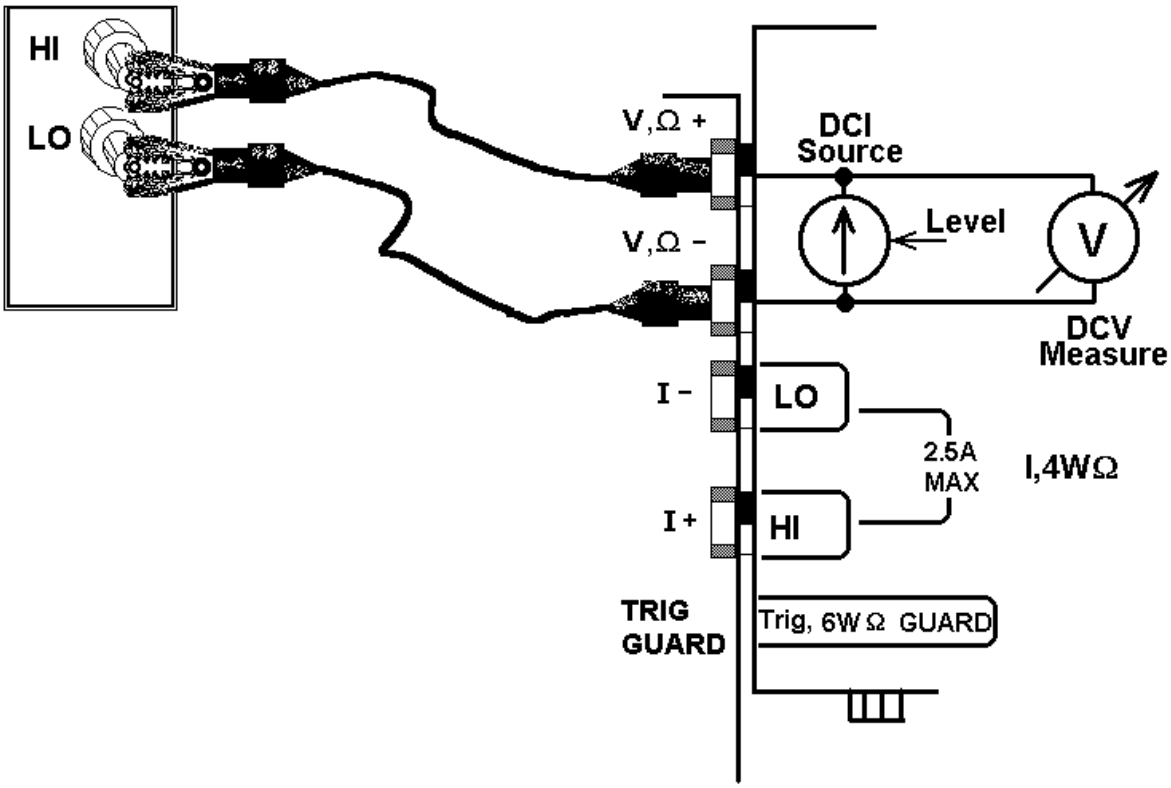


Figure 4-12. Sourcing DC current and measuring voltage in the two wire configuration. This function can be used for semiconductor parametric tests.

4.12.5 Source Voltage - Measure Current

To source DC Voltage while measuring the current through a load, connect an external shunt resistor as described in Figure 4-6. The details in section 4.3.4, for Leakage measurements are also applicable to Sourcing Voltage and Measuring Current operations.

5.0 DMM Windows Interface

5.1 Distribution Files

The main directory of the distribution diskette contains the Microsoft® Windows™ MODEL 1004 DMM software. Before installing the DMM or software, read the "Quick Install" page carefully. To install this software, enter the command "A:SETUP" in the "Run Program" menu of the Windows File Manager; or double-click on the SETUP.EXE file name from the File Explorer Tool Manager window. Most files on this diskette are compressed, and must be installed using the SETUP program.

The MODEL 1004 DLL is a protected-mode Microsoft® Windows™ DLL that will control the Ascor DMM. It is provided with a sample Visual Basic™ front-panel application to demonstrate the DMM and the interface to the DLL. Check the README.TXT file for more information about the files contained on the diskette. Some important files to note are:

File	Description
SM40CAL.DAT	Configuration file containing calibration information for each DMM. Do not write into this file unless you are performing an external calibration! This file is normally placed at the C:\ root directory by the setup program, and should be left there. It may contain calibration records for several DMMs.
MODEL 100432.LIB	The Windows import library. Install in a directory pointed to by your **LIB** environment variable.
MODEL 100432.DEF	MODEL 1004 driver DLL module definition file.
MODEL 100432.DLL	The 32 bit driver DLL. This should be installed either in your working directory, in the Windows system directory, or in a directory on your **PATH**. The installation program installs this file in your Windows system directory (usually **C:\WINDOWS\SYSTEM for Win98/95 or at C:\WINNT\SYSTEM32 for Windows NT**).
MODEL 100432.H	Driver header file. Contains the definitions of all the DMM's function prototypes for the DLL, constant definitions, and error codes. Install in a directory pointed to by your **INCLUDE** environment variable.
UserDMM.H	Header file containing all of the necessary DMM's function, range, rate definitions to be used with the various measure and source functions.
Msvbvm50.dll	Visual Basic run-time interpreter. Usually, install in your C**:\WINDOWS\SYSTEM** (or equivalent) directory. If it is not already installed, run Msvbvm50.exe for proper extraction and registration.
MODEL 1005.vbw	Visual Basic project file
MODEL 1005.frx	Visual Basic binary form file
MODEL 1005.frm	Visual Basic file with main form
MODEL 1005.vbp	Visual Basic project file
2044glbl.bas	Visual Basic file with all global DMM declarations

File	Description
MODEL 1005.exe	Visual Basic DMM control panel executable
Msvcrt.dll	System file. Installs in your **C:\WINDOWS\SYSTEM** directory.

Important Note about the SM40CAL.DAT file:

The file **SM40CAL.DAT** contains calibration information for each DMM, and determines the overall analog performance for that DMM. You must not alter this file unless you are performing an external calibration of the DMM. This file may contain multiple records for more than one DMM. Each record starts with a header line, followed by calibration data.

```
card_id  10123    type 2044 calibration_date 06/15/1999
ad          ; A/D compensation
72.0        20.0
vdc         ; VDC 330mV, 3.3V, 33V, 330V ranges. 1st entry is Offset the 2nd is gain parameters
-386.0 0.99961
-37.0 .999991
-83.0 0.999795
-8.8  1.00015
vac         ; VAC 1st line - DC offset. Subsequent lines: 1st entry is Offset the 2nd is gain, 3rd freq. comp
5.303       ; starting with the 330mV range, and last line is for the 250V range.
0.84                1.015461            23
0.0043              1.0256              23
0.0                 1.02205             0
0.0                 1.031386            0
idc         ; IDC 3.3mA to 2.5A ranges. 1st entry is offset, 2nd is gain parameter
-1450.0   1.00103
-176.0   1.00602
-1450.0   1.00482
-176.0   1.0
iac         ; IAC 3.3mA to 2.5A ranges, offset and gain
1.6   1.02402
0.0   1.03357
1.69   1.00513
0.0   1.0142
2w-ohm  ; Ohms 33, 330, 3.3k,...,330Meg ranges, offset and gain
12700.0             1.002259            ;in the MODEL 1004, the 1st and last lines are placeholders
1256.0              1.002307
110.0               1.002665
0.0                 1.006304
0.0                 1.003066
0.0                 1.001848
0.0                 0.995664
```

0.0 1.00030

…

.

The first line identifies the DMM and the calibration date. The "card-id" is stored in ROM on each DMM. During initialization the driver uses the information from the **DMM.CFG** file to identify where the DMM is located in I/O space, reads the "card-id" and "calibration_date", and then reads the corresponding calibration information from the **SM40CAL.DAT** file.

During initialization (**DMMInit()**), the driver reads various parameters such as DMM type (MODEL 1004/42/44), and serial number, and then reads the corresponding calibration information from the **SM40CAL.DAT** file.

The **DMMInit()** function reads the information from these files to initialize the DMM. DMM**Init** accepts parameters that are the names of these files. A qualified technician may modify individual entries in the calibration file, then reload them using the **DMMLoadCalFile** command.

5.6 Windows Command Language

The following section contains detailed descriptions of each function of the Windows command language.Those commands that pertain to only the MODEL 1004 are indicated. Most functions return an error code. The code can either be retrieved as a string usinge **DMMErrString** function, or looked up in the **SM204032.H** header file. The **UserDMM.H** file contains all the pertinant definitions for the DMM ranes functions etc. The following description for the various functions is based on "C" function declarations. Keep in mind that the Windows DLL containing these functions assumes all **int** values to be windows 32bit integers (corresponds to VisualBasic long type). TRUE is equal to 1 and FALSE to 0 (which is different from VisualBasic).

The following conversion chart shows the relationship between the name of the DMM Native Driver Functions and the VXI *Plug&Play* Dmm Driver Functions,

VXI *PLUG&PLAY* DMM DRIVER FUNCTIONS	**DMM NATIVE DRIVER FUNCTIONS**
as1005_DMMArmAnalogTrigger	*DMMArmAnalogTrigger*
as1005_DMMArmTrigger	*DMMArmTrigger*
as1005_DMMBurstBuffRead	*DMMBurstBuffRead*
as1005_DMMBurstRead	*DMMBurstRead*
as1005_DMMCalibrate	*DMMCalibrate*
as1005_DMMClearMinMax	*DMMClearMinMax*
as1005_DMMDelay	*DMMDelay*
as1005_DMMDisableTrimDAC	*DMMDisableTrimDAC*
as1005_DMMDisArmTrigger	*DMMDisArmTrigger*
as1005_DMMDutyCycleStr	*DMMDutyCycleStr*
as1005_DMMFrequencyStr	*DMMFrequencyStr*
as1005_DMMGetCalDate	*DMMGetCalDate*
as1005_DMMGetdB	*DMMGetdB*
as1005_DMMGetdBStr	*DMMGetdBStr*
as1005_DMMGetDeviation	*DMMGetDeviation*
as1005_DMMGetDeviatStr	*DMMGetDeviatStr*
as1005_DMMGetFuncRange	*DMMGetFuncRange*

VXI *PLUG&PLAY* DMM DRIVER FUNCTIONS	DMM NATIVE DRIVER FUNCTIONS
as1005_DMMGetFunction	*DMMGetFunction*
as1005_DMMGetGrdVer	*DMMGetGrdVer*
as1005_DMMGetHwVer	*DMMGetHwVer*
as1005_DMMGetID	*DMMGetID*
as1005_DMMGetManDate	*DMMGetManDate*
as1005_DMMGetMax	*DMMGetMax*
as1005_DMMGetMaxStr	*DMMGetMaxStr*
as1005_DMMGetMin	*DMMGetMin*
as1005_DMMGetMinStr	*DMMGetMinStr*
as1005_DMMGetRange	*DMMGetRange*
as1005_DMMGetRate	*DMMGetRate*
as1005_DMMGetSlot	*DMMGetSlot*
as1005_DMMGetSourceFreq	*DMMGetSourceFreq*
as1005_DMMGetType	*DMMGetType*
as1005_DMMGetVer	*DMMGetVer*
as1005_DMMInit	*DMMInit*
as1005_DMMIsAutoRange	*DMMIsAutoRange*
as1005_DMMIsInitialized	*DMMIsInitialized*
as1005_DMMIsRelative	*DMMIsRelative*
as1005_DMMLoadCalFile	*DMMLoadCalFile*
as1005_DMMOpenTerminalCal	*DMMOpenTerminalCal*
as1005_DMMPeriodStr	*DMMPeriodStr*
as1005_DMMPolledRead	*DMMPolledRead*
as1005_DMMPolledReadCmd	*DMMPolledReadCmd*
as1005_DMMPolledReadStr	*DMMPolledReadStr*
as1005_DMMRead	*DMMRead*
as1005_DMMReadBuffer	*DMMReadBuffer*
as1005_DMMReadBufferStr	*DMMReadBufferStr*
as1005_DMMReadCrestFactor	*DMMReadCrestFactor*
as1005_DMMReadDutyCycle	*DMMReadDutyCycle*

VXI *PLUG&PLAY* DMM DRIVER FUNCTIONS	DMM NATIVE DRIVER FUNCTIONS
as1005_DMMReadFrequency	*DMMReadFrequency*
as1005_DMMReadInductorQ	*DMMReadInductorQ*
as1005_DMMReadMeasurement	*DMMReadMeasurement*
as1005_DMMReadMedian	*DMMReadMedian*
as1005_DMMReadNorm	*DMMReadNorm*
as1005_DMMReadPeakToPeak	*DMMReadPeakToPeak*
as1005_DMMReadPeriod	*DMMReadPeriod*
as1005_DMMReadStr	*DMMReadStr*
as1005_DMMReadTotalizer	*DMMReadTotalizer*
as1005_DMMReadWidth	*DMMReadWidth*
as1005_DMMReady	*DMMReady*
as1005_DMMSetACVSource	*DMMSetACVSource*
as1005_DMMSetAutoRange	*DMMSetAutoRange*
as1005_DMMSetBuffTrigRead	*DMMSetBuffTrigRead*
as1005_DMMSetCapsMeasure	*DMMSetCapsMeasure*
as1005_DMMSetCompThreshold	*DMMSetCompThreshold*
as1005_DMMSetCounterRng	*DMMSetCounterRng*
as1005_DMMSetDCISource	*DMMSetDCISource*
as1005_DMMSetDCVSource	*DMMSetDCVSource*
as1005_DMMSetFuncRange	*DMMSetFuncRange*
as1005_DMMSetFunction	*DMMSetFunction*
as1005_DMMSetInductFreq	*DMMSetInductFreq*
as1005_DMMSetRange	*DMMSetRange*
as1005_DMMSetRate	*DMMSetRate*
as1005_DMMSetRelative	*DMMSetRelative*
as1005_DMMSetRTD	*DMMSetRTD*
as1005_DMMSetSourceMode	*DMMSetSourceMode*
as1005_DMMSetSynchronized	*DMMSetSynchronized*
as1005_DMMSetTempUnits	*DMMSetTempUnits*
as1005_DMMSetTrigRead	*DMMSetTrigRead*

VXI *PLUG&PLAY* DMM DRIVER FUNCTIONS	DMM NATIVE DRIVER FUNCTIONS
as1005_DMMSetTrimDAC	*DMMSetTrimDAC*
as1005_DMMStartTotalizer	*DMMStartTotalizer*
as1005_DMMStopTotalizer	*DMMStopTotalizer*
as1005_DMMTerminate	*DMMTerminate*
as1005_DMMTrigger	*DMMTrigger*
as1005_DMMWidthStr	*DMMWidthStr*

DMMArmAnalogTrigger

MODEL 1004 ☑ MODEL 1005 ☑

Description Arm DMM for analog level trigger operation.

int DMMArmAnalogTrigger(**int** *nDmm*, **int** *iSamples*, **double FAR ***_dThresh_)

Remarks This function is usable for VDC, VAC, Ohms, IAC, and IDC. Setup the MODEL 1004 for analog level trigger operation. Following reception of this command the DMM makes measurements continuously, waiting for a value, which exceeds the threshold, *dThresh.* When this occurs, a trigger is produced with identical processing as in **DMMArmTrigger()**. Threshold crossing sense is determined by the first measurement following the call of **DMMArmAnalogTrigger()**. If that measurement is lower than the set threshold, *dThresh*, subsequent measurements greater than *dThresh* will trigger the DMM. If the first measurement is greater than *dThresh*, subsequent measurements smaller than *dThresh* will trigger. For example, if *dThresh* is 2.00000 V and the first reading after arming the DMM is 2.500000 V, then 1.999999 V (or smaller) will trigger the DMM. On the other hand, if *dThresh* is 1.000000 V and the first reading after arming the DMM is 0.500000 V, then 1.000001 V (or greater) will trigger the DMM.

The *dThresh* value is in base units, and must be within the DMM range setting. For example, in the 330 mV range, *dThresh* must be within -0.330000 and +0.330000. In the 33 kΩ, range *dThresh* must be between 0.0 and 33.0e3.

Following an analog level trigger event, the DMM makes *iSamples* readings at the set function, range, and reading rate, and stores them in an internal buffer. Autoranging is not allowed when using **DMMAnalogTrigger()**. Between the time the **DMMArmAnalogTrigger()** is issued and the time the buffer is read, no other command should be sent to the DMM. One exception is the **DMMDisArmTrigger** command.

Use the **DMMReady()** to monitor when the DMM is ready. When ready, you can read up-to *iSamples*, using **DMMReadBuffer** or **DMMReadBufferStr** functions. Once **DMMReady()** returns **TRUE**, it should not be used again prior to reading the buffer, since it prepares the buffer for reading when it detects a ready condition.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
iSamples	**int** The number of samples the DMM takes following a trigger pulse. This number must be between 1 and 64, inclusive.
dThresh	**double FAR** Analog level trigger threshold value

Return Value The return value is one of the following constants.

Value	Meaning
DMM_OKAY	Operation successfully terminated
Negative value	Error code.

Example

```
double Buffer[64];

      DMMArmAnalogTrigger(0,64,1.5);
while( ! DMMReady(0));

      for(i=0; i < 64 ; i++)

            j = DMMReadBuffer(0, &Buffer[i]);
```

DMMArmTrigger

MODEL 1004 ☑ MODEL 1005 ☑

Description Arm DMM for external trigger operation.

int DMMArmTrigger(**int** *nDmm*, **int** *iSamples*)

Remarks Setup the MODEL 1004 for external hardware trigger operation. Following reception of this command the DMM enters a wait state. After reception of an external trigger pulse, the DMM makes *iSamples* readings at the set function, range, and reading rate; and stores them in an internal buffer. No autoranging is allowed for external trigger operation. Between the time the **DMMArmTrigger()** is issued and the time the buffer is read, no other command should be sent to the DMM. One exception is the **DMMDisarmTrigger** command. This function is usable for VDC, VAC, Ohms, IAC, and IDC.

Use the **DMMReady()** to monitor when the DMM is ready (following trigger and the reading of *iSamples*). When ready, you can read up to *iSamples*, using **DMMReadBuffer** or **DMMReadBufferStr** functions. Once **DMMReady()** returns TRUE, it should not be used again prior to reading the buffer, since it prepares the buffer for reading when it detects a ready condition.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
iSamples	**int** The number of samples the DMM takes following a trigger pulse. This number must be between 1 and 64, inclusive.

Return Value The return value is one of the following constants.

Value	Meaning
DMM_OKAY	Operation successfully terminated
Negative Value	Error code.

Example

```
double Buffer[64];

      DMMArmTrigger(0,64);
while( ! DMMReady(0));

      for(i=0; i < 64 ; i++)

            j = DMMReadBuffer(0, &Buffer[i]);
```

DMMBurstBuffRead

MODEL 1004 ☑ MODEL 1005 ☑

Description Setup the DMM for Triggered operation.

int DMMBurstBuffRead(int *nDmm***, int** *iSettle***, int** *iSamples***)**

Remarks Following reception of this command the DMM enters a burst read mode, where it takes *iSettle* + 1 readings at the set measurement function, range, and reading rate; and saves the last reading to the on-board buffer. This process repeats for *iSamples*. No other DMM command should be issued to the DMM until it completes the operation, and the buffer is read. One exception is the **DMMDisarmTrigger** command, which terminates the process. No autoranging is allowed in this mode. This function is usable for VDC, VAC, Ohms, IAC, IDC, and RTD measurements. Measurement rate should be set to 10rps or higher. The total time it takes to complete this process is equal to *iSamples* * (*iSettle* + 1) / (measurement rate).

Use the **DMMReady** to monitor if the has completed the operation, and is ready. When ready, read up to *iSamples*, using **DMMReadBuffer** or **DMMReadBufferStr** functions. Once **DMMReady** returns TRUE, it should not be used again until the buffer is read, since it clears some flags in preparation for buffer reading when it detects a ready condition.

Parameter	Type/Description
iDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
iSettle	**int** The number of setteling measurements, prior to read value. Must be set between 0 and 250. Recommanded value is 4.
iSamples	**int** The number of samples the DMM takes following the same number of trigger pulses. This number must be between 1 and 64, inclusive.

Return Value The return value is one of the following constants.

Value	Meaning
DMM_OKAY	Operation successfully terminated
Negative Value	Error code.

Example

```
double Buffer[50];

      DMMBurstBuffRead(0, 4, 50); // 4 setteling readings for each //
measurement, and take 50 readings
while( ! DMMReady(0) );       // wait for completion

      for(i=0; i < 50 ; i++)        // read 64 readings from DMM's // on-
board buffer

            j = DMMReadBuffer(0, &Buffer[i]);
```

DMMBurstRead

MODEL 1004 ☑ MODEL 1005 ☑

Description Setup the DMM for mutiple readings operation, sending back measurements as they come.

int DMMBurstRead(int *nDmm***, int** *iSettle***, int** *iSamples***)**

Remarks Set the DMM to take multiple measurements, sending readings back to the PC. This function is similar to the **DMMSetTrigRead** function, with the exception that it does not wait for a hardware trigger to start making measurements. Following reception of this command the DMM enters a burst read mode, where it takes *iSettle* + 1 readings at the set measurement function, range, and reading rate; and sends the last reading to the PC. This process repeats for *iSamples*. Following the issue of this command, and until *iSamples* measurements are read, it is necessary to read the samples from the DMM using the **DMMReadMeasurement** command as fast as they become available. This will prevent an Overrun communication error, which is an indication that the rate at which measurements are read from the bus do not keep up with the DMM transmission. The DMM has five readings Fifo to lessen this problem. No autoranging is allowed in this mode. This function is usable for VDC, VAC, Ohms, IAC, IDC, and RTD measurements. Measurement rate should be set to 10rps or higher. The total time it takes to complete this process is equal to *iSamples* * (*iSettle* + 1) / (measurement rate).

Use the **DMMReadMeasurement** to monitor when reading becomes available, and to read the data. Read as many samples as *iSamples* to guarantee proper conclusion of this capture process.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
iSettle	**int** The number of setteling measurements, prior to read value. Must be set between 0 and 250. Recommanded value is 4.
iSamples	**int** The number of samples the DMM takes following the same number of trigger pulses. This number must be between 1 and 32,000, inclusive.

Return Value The return value is one of the following constants.

Value	Meaning
DMM_OKAY	Operation successfully terminated
Negative Value	Error code.

Example

```
double Reading[250];

      DMMBurstRead(0, 10, 250); // settle 10 reads., 250 samples for(i=0; i < 250 ; i++)  // read 250 meas. as they come

            while( ! DMMReadMeasurement(0 , Reading[i]) );
```

DMMCalibrate

MODEL 1004 ☑ MODEL 1005 ☑

Description Internally calibrate the DMM.

int DMMCalibrate(int *nDmm*)

Remarks This function re-calibrates the DMM, and returns it to the current operating mode.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.

Return Value The return value is one of the following constants.

Value	Meaning
DMM_OKAY	DMM is OK.
Negative Value	Error

Example

```
status = DMMCalibrate(0); /* a quick internal cal.*/
```

Comments This performs an internal DMM calibration and is the same as the **S-Cal** command in the VB Control Panel. It is not related to the external calibration represented in the **SM40CAL.DAT** file.

DMMClearMinMax

MODEL 1004 ☑ MODEL 1005 ☑

Description Clears the Min/Max storage.

int DMMGetMin(int *nDmm***)**

Remarks This function clears the Min/Max values, and initiates a new Min/Max accumulation. See **DMMGetMin** for more details.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.

Return Value Integer error code.

Value	Meaning
DMM_OKAY	Operation successfully completed.
Negative Value	Error code

Example

```
int status = DMMClearMinMax(0);
```

DMMDelay

MODEL 1004 ☑ MODEL 1005 ☑

Description Wait for a given time.

int DMMDelay(double *dTime*)

Remarks Delay of *dTime* seconds. *dTime* must be a positive double number between 0.0 and 100.0 seconds.

Parameter	Type/Description
dTime	**double** Delay time in seconds.

Return Value The return value is one of the following constants.

Value	Meaning
DMM_OKAY	Operation successfully terminated
Negative Value	Error code

Example

```
DMMDelay(1.2); /* wait for 1.2 Sec */
```

DMMDisableTrimDAC

MODEL 1004 ☐ MODEL 1005 ☑

Description Terminate the operation of the Trim DAC.

int DMMDisableTrimDAC(int *nDmm*)

Remarks This function disables the Trim DAC. Since usage of the Trim DAC consumes a lot of the on-board microcontroller's resources it must be turned off with this function when not in use. See **DMMSetTrimDAC()**, **DMMSetDCVSource(),** and **DMMSetACVSource()** for more details.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.

Return Value Integer error code.

Value	Meaning
DMM_OKAY	Operation successfully completed.
Negative Value	Error code

Example

```
DMMDisableTrimDAC(0); // Remove Trim DAC from operation
```

DMMDisArmTrigger

MODEL 1004 ☑ MODEL 1005 ☑

Description Abort trigger operation.

int DMMDisArmTrigger(**int** *nDmm*)

Remarks This function sends the DMM a trigger termination command. If the DMM is waiting for a trigger, it will exit the wait mode, and be ready for a new operation. It can be used following an external hardware or analog level trigger arm command (**DMMArmAnalogTrigger(), DMMArmTrigger(),** or **DMMTrigger()**). It can be used with no limitation.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.

Return Value Integer error code

Value	Meaning
DMM_OKAY	Operation successfully completed.
Negative Value	Error code

DMMDutyCycleStr

MODEL 1004 ☐ MODEL 1005 ☑

Description Return percent duty cycle of an AC signal in string format.

int DMMDutyCycleStr(**int** *nDmm*, **LPSTR** *lpszReading*)

Remarks This function is the string version of **DMMReadDutyCycle()**. The measurement result is stored at the location pointed to by *lpszReading*. See **DMMReadDutyCycle()** for more details.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
lpszReading	**LPSTR** Points to a buffer (at least 64 characters long) to hold the result.

Return Value The return value is one of the following constants.

Value	Meaning
DMM_OKAY	Valid return.
Negative Value	Error code

Example

```
char cBuf[64]; int status = DMMDutyCycleStr(0, cBuf);
```

DMMFrequencyStr

MODEL 1004 ☐ MODEL 1005 ☑

Description Return the next DMM frequency reading, formatted for printing.

int DMMFrequencyStr(**int** *nDmm*, **LPSTR** *lpszReading*)

Remarks This function makes frequency measurement and returns the result as a string formatted for printing. The print format is fixed to six digits plus units, e.g., 05.001 Hz. If the DMM is in autorange, be certain to take an amplitude reading before using this command. It may take several calls to **DMMFrequencyStr()** to get the measured frequency, because the DMM frequency counter uses a frequency ranging scheme which gets activated only when a frequency or period reading function is received. If the previously measured frequency was 1 Hz and the frequency being measured is 300 kHz (or vise versa), it might take as many as six calls to **DMMFrequencyStr()** or any of the other frequency measurement functions, to read the correct frequency. This function is a **Secondary** function, which requires the DMM to be in either VAC or IAC function and at the appropriate range.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
lpszReading	**LPSTR** Points to a buffer (at least 64 characters long) to hold the converted result.

Return Value The return value is one of the following constants.

Value	Meaning
DMM_OKAY	Operation successfully completed.
DMM_CNT_RNG	Frequency counter is over or under range.
Negative Value	Error code

Example

```
char cBuf[64];
int status;
status = DMMFrequencyStr(0, cBuf);
```

DMMGetCalDate

MODEL 1004 ☑ MODEL 1005 ☑

Description Return the calibration date string from the DMM.

int DMMGetCalDate(**int** *nDmm*, **LPSTR** *lpszCalDate*)

Remarks This function reads the calibration date string from the structure.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
lpszCalDate	**LPSTR** Points to a buffer (at least 64 characters long) to hold the cal date string.

Return Value The return value is one of the following constants.

Value	Meaning
any positive number	Length of the date string
Negative number	Error code

Example

```
char cBuf[64];
int status;
status = DMMGetCalDate(0, cBuf);
```

DMMGetdB

MODEL 1004 ☑ MODEL 1005 ☑

Description Get dB deviation from the reading at the time relative was activated.

int DMMGetdB(int *nDmm*, **double FAR ***_lpdDev_**)**

Remarks This function returns a double floating value that is the dB deviation relative to the reading made just before the relative function was activated. This function is useful in determining measurement errors in dB. It can be used for bandwidth measurements or DC evaluation.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
lpdDev	**double FAR *** Pointer where the dB value is to be saved.

Return Value Integer error code.

Value	Meaning
DMM_OKAY	Operation successfully completed.
Negative Value	Error code

Example

```
double FAR dB; int status = DMMGetdB(0, &dB);
```

DMMGetdBStr

MODEL 1004 ☑ MODEL 1005 ☑

Description Get dB deviation from the reading at the time relative was activated.

int DMMGetdBStr(int *nDmm***, LPCSTR** *lpszDB***)**

Remarks This function is the same as the **DMMGetdB()**, with the exception that it returns a string. See **DMMGetdB()** for more details.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
lpszDB	**LPCSTR** Points to a buffer (at least 64 characters long) to hold the result. The return value will consist of a leading sign, a floating-point, and a 'dB' unit specifier.

Return Value Integer string length if successful, or an error code.

Value	Meaning
Negative Value	Error code

Example

```
char cBuf[64]; int strLength = DMMGetdBStr(0, cBuf);
```

DMMGetDeviation

MODEL 1004 ☑ MODEL 1005 ☑

Description Get percent deviation from the reading at the time relative was activated.

int DMMGetDeviation(int *nDmm***, double FAR ****lpdDev***)**

Remarks This function returns a double floating value that is the percent deviation relative to the reading made just before the relative function was activated. This function is useful in quantifying measurement errors. It can be used for bandwidth measurements or DC evaluation, or percent variation of a device under test over temperature.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
lpdDev	**double FAR *** Pointer where the deviation value is to be saved.

Return Value Integer error code.

Value	Meaning
DMM_OKAY	Operation successfully completed.
Negative Value	Error code

Example

```
double FAR error;
int status = DMMGetDeviation(0, &error);
```

DMMGetDeviatStr

MODEL 1004 ☑ MODEL 1005 ☑

Description Get percent deviation from the reading at the time relative was activated.

int DMMGetDeviatStr(int *nDmm***, LPCSTR** *lpszDev***)**

Remarks This function is the same as the **DMMGetDeviation()**, with the exception that it returns a string. See **DMMGetDeviation()** for more details.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
lpszDev	**LPCSTR** Points to a buffer (at least 64 characters long) to hold the result. The return value will consist of a leading sign a floating-point, and a % unit specifier.

Return Value Integer string length if successful, or an error code.

Value	Meaning
Negative Value	Error code

Example

```
char cBuf[64];
int strLength = DMMGetDeviatStr(0, cBuf);
```

DMMGetFuncRange

MODEL 1004 ☑ MODEL 1005 ☑

Description Get DMM range code.

int DMMGetFuncRange(int *nDmm*)

Remarks This function returns the combined DMM function/range code. See **UserDMM.h** for the complete set of codes.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.

Return Value Integer value corresponding to the currently set DMM function/range, or an error code. The following are a few examples of the returned value.

Value	Meaning
VDC_300mV	First and lowest VDC range (330 mV)
VDC_300V	Highest VDC range.
VAC_300mV	First and lowest VAC range (330 mV)
VAC_300V	Highest VAC range.
IAC_3mA	First and lowest IAC range (3.3 mA)
IAC_300mA	Highest IAC range.
IDC_3mA	First and lowest IDC range is 3.3 mA.
IDC_300mA	Highest IDC range.
OHM_2W_300	First 2-wire Ohms range is 330 Ohms.
OHM_2W_300K	Fourth 2-wire Ohms range is 330 k.
OHM_2W_30MEG	The highest 2-wire Ohms range is 33 MOhms.
OHM_4W_3K	Second 4-wire Ohms range is 3.3 k.
OHM_4W_300K	Fourth 4-wire Ohms range is 330 k
Negative Value	Error code

Example

```
if(DMMGetFuncRange(0) == VDC_300mV) printf("Lowest VDC range selected");
```

DMMGetFunction

MODEL 1004 ☑ MODEL 1005 ☑

Description Get DMM function code.

int DMMGetFunction(int *nDmm*)

Remarks This function returns the DMM function code.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.

Return Value Integer value corresponding to the current function, or an error code.

Value	Meaning
VDC	Volts DC
VAC	Volts AC
IAC	AC current
IDC	DC current
OHMS2W	2-wire Ohms
OHMS4W	4-wire Ohms
DIODE	Diode Characterization
TEMP_LCL	Local Board Temperature
CAPS	Capacitance
RTD	Temperature with RTD
VDC_SRC	VDC Source
VAC_SRC	VAC Source
IDC_SRC	IDC Source
LEAKAGE	Leakage
INDUCTANCE	Inductance
VDCSRC_IDCSNS	Source Voltage Measure Current
Negative Value	Error code

Example

```
if(DMMGetFunction() == VDC) printf("VDC Function selected");
```

DMMGetGrdVer

MODEL 1004 ☑ MODEL 1005 ☑

Description Get DMM firmware version.

int DMMGetGrdVer(int *nDmm*)

Remarks This function returns the DMM firmware version of the on-board controller.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.

Return Value Integer value. The return value is the version value or an error code.

Value	Meaning
Positive Value	Version
Negative Value	Error code

Example

```
firmwarever = DMMGetGrdVer(0);
```

DMMGetHwVer

MODEL 1004 ☑ MODEL 1005 ☑

Description Get the hardware version of the DMM.

int DMMGetHwVer(int *nDmm*)

Remarks This function returns the DMM hardware version. A returned value of 0 corresponds to Rev_, 1 corresponds to Rev_A, 2 to Rev_B, etc.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.

Return Value DMM hardware code or an error code.

Value	Meaning
Positive value	Hardware version code
Negative Value	Error code

Example

```
int HWVer = DMMGetHwVer(0);
```

DMMGetID

MODEL 1004 ☑ MODEL 1005 ☑

Description Get DMM ID code.

int DMMGetID(**int** *nDmm*)

Remarks This function returns the DMM identification code. Each DMM has a unique ID code that must match the calibration file **card_ID** field in **SM40CAL.DAT.**

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.

Return Value Integer value card ID code (serial number) or an error code.

Value	Meaning
DMM_E_DMM	Invalid DMM number.

Example

```
int id = DMMGetID(0);
```

DMMGetManDate

MODEL 1004 ☑ MODEL 1005 ☑

Description Get Manufacturing date stamp from the DMM hardware.

int DMMGetManDate(int *nDmm*, **int ****month*, **int ****day*, **int ****year*)

Remarks This function returns the DMM manufacturing date, which is read from the hardware. The month, day, and year are returned as integers. This is used to track the DMM to a specific manufacturing date.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
month	**int *** A pointer to an integer where the month is stored.
day	**int *** A pointer to an integer where the day is stored.
year	**int *** A pointer to an integer where the year is stored.

Return Value Integer error code.

Value	Meaning
DMM_OKAY	Operation was successful.
DMM_E_DMM	Invalid DMM number.

Example

```
int month, day, year, status;
status = DMMGetManDate(0, &month, &day, &year);
```

DMMGetMax

MODEL 1004 ☑ MODEL 1005 ☑

Description Get the maximum reading history.

int DMMGetMax(int *nDmm***, double FAR ***lpdMax***)**

Remarks This function returns a double floating value that is the maximum (of the Min/Max function) value since either a function change, range change, or call to the **DMMClearMinMax** function was made. This is only applicable to **Primary** read functions (those that are read using **DMMRead()**, **DMMReadStr(),** or **DMMReadNorm()**). This value is updated every time one of those functions is used.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
lpdMax	**double FAR *** Pointer where the Max value is to be saved.

Return Value Integer error code.

Value	Meaning
DMM_OKAY	Operation successfully completed.
Negative Value	Error code

Example

```
double FAR Mx; int status = DMMGetMax(0, &Mx);
```

DMMGetMaxStr

MODEL 1004 ☑ MODEL 1005 ☑

Description Returns the maximum as a formatted string.

int DMMGetMaxStr(**int** *nDmm*, **LPSTR** *lpszReading*)

Remarks This function is the string version of **DMMGetMax**. It returns the result as a string formatted for printing. The print format is determined by the range and function. See **DMMGetMax()** for more details.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
lpszReading	**LPSTR** Points to a buffer (at least 64 characters long) to hold the result.

Return Value The return value is one of the following constants, or the string length if OK.

Value	Meaning
DMM_OKAY	Valid return.
Negative Value	Error code

Example

```
char cBuf[64];
int status = DMMGetMaxStr(0, cBuf);
```

DMMGetMin

MODEL 1004 ☑ MODEL 1005 ☑

Description Get the minimum reading history.

int DMMGetMin(int *nDmm***, double FAR ***lpdMin**)**

Remarks This function returns a double floating value that is the minimum (of the Min/Max function) value since either a function change, range change, or a call to the **DMMClearMinMax**() function was made. This is only applicable to **Primary** read functions (those that are read using **DMMRead()**, **DMMReadStr()** or **DMMReadNorm()**). This value is updated every time one of those functions is used.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
lpdMin	**double FAR *** Pointer where the Min value is to be saved.

Return Value Integer error code.

Value	Meaning
DMM_OKAY	Operation successfully completed.
Negative Value	Error code

Example

```
double FAR Min; int status = DMMGetMin(0, &Min);
```

DMMGetMinStr

MODEL 1004 ☑ MODEL 1005 ☑

Description Returns the minimum as a formatted string.

int DMMGetMinStr(**int** *nDmm*, **LPSTR** *lpszReading*)

Remarks This function is the string version of **DMMGetMin()**. It returns the result as a string formatted for printing. The print format is determined by the range and function. See **DMMGetMin()** for more details.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
lpszReading	**LPSTR** Points to a buffer (at least 64 characters long) to hold the result.

Return Value The return value is one of the following constants, or the string length if OK.

Value	Meaning
DMM_OKAY	Valid return.
Negative Value	Error code

Example

```
char cBuf[64];
int status = DMMGetMinStr(0, cBuf);
```

DMMGetRange

MODEL 1004 ☑ MODEL 1005 ☑

Description Get DMM range code.

int DMMGetRange(int *nDmm*)

Remarks This function returns the DMM range code.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.

Return Value Integer value corresponding to the currently set DMM range, or an error code.

Value	Meaning
0	First and lowest range (330 mV, 33 Ohms, 3.3 mA)
1	Second range (3.3 V, 330 Ohm, 33 mA)
2	Third range
3	Fourth range
4	Fifth range (330 kOhm)
5	Sixth range (3.3 Mohm, 3.3 H)
6	Seventh range (33 MOhm)
7	Eighth range (330 MOhm)
Negative Value	Error code

Example

```
int id;
if(DMMGetRange(0) == 0) printf("Lowest range selected");
```

DMMGetRate

MODEL 1004 ☑ MODEL 1005 ☑

Description Get DMM reading rate.

int DMMGetRate(int *nDmm***, double FAR *** *lpdRate***)**

Remarks This function returns a double floating rate in readings per second.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
lpdRate	**double FAR *** Pointer to double where the rate is to be saved.

Return Value Integer error code.

Value	Meaning
Negative Value	Error code

Example

```
int status; double FAR rate;
status = DMMGetRate(0, &rate);
```

DMMGetSlot

MODEL 1004 ☑ MODEL 1005 ☑

Description Get the slot number where the DMM is plugged in.

int DMMGetSlot(int *nDmm*)

Remarks This function returns the slot number of the VXI chassis where the DMM is located.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.

Return Value DMM slot number or error code.

Value	Meaning
Positive value	Slot number
Negative Value	Error code

Example

```
int DMM0Slot = DMMGetSlot(0);
```

DMMGetSourceFreq

MODEL 1004 ☐ MODEL 1005 ☑

Description Get the currently set ACV Source frequency.

int DMMGetSourceFreq(int *nDmm***, double FAR *** *lpdFreq***)**

Remarks This function returns a double floating value that is the currently set ACV source frequency of the MODEL 1005. It can be used to display or verify the default frequency of the stimulus for the various Inductance measurement ranges.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
lpdFreq	**double FAR *** Pointer where the frequency value is to be saved.

Return Value Integer error code.

Value	Meaning
DMM_OKAY	Operation successfully completed.
Negative Value	Error code

Example

```
double FAR f; int status = DMMGetSourceFreq(0, &f);
```

DMMGetType

MODEL 1004 ☑ MODEL 1005 ☑

Description Get the type of the DMM.

int DMMGetType(int *nDmm*)

Remarks This function returns the DMM type.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.

Return Value DMM type Integer or an error code.

Value	Meaning
2040	MODEL 1004 is at nDmm slot
2044	MODEL 1005 is at nDmm slot
Negative Value	Error code

Example

```
int type = DMMGetType(0);
```

DMMGetVer

MODEL 1004 ☑ MODEL 1005 ☑

Description Get DMM software driver version.

int DMMGetVer(int *nDmm*, **double FAR *** *lpfResult*)

Remarks This function returns the DMM software driver version, which is a double floating value.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
lpfResult	**double FAR *** Pointer to the location which holds the version.

Return Value Integer error code.

Value	Meaning
Negative Value	Error code

Example

```
int status; double ver;
status = DMMGetVer(0, &ver);
```

DMMInit

SM2040 ☑ SM2042 ☑ SM2044 ☑

Description Initialize a DMM.

int DMMInit(**int** *nDmm*, **LPCSTR** *lpszCal*)

Remarks This function must be the first function to be executed. It opens the driver for the specified DMM. The first DMM becomes 0, the second 1, etc. It also initializes the DMM hardware and software and reads the appropriate calibration record for the respective DMM from the file specified by *lpszCal*.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
lpszCal	**LPCSTR** Points to the name of the file containing the calibration constants for the DMM. Calibration information is normally read from the file named **SM40CAL.DAT** located in the current directory.

Return Value The return value is one of the following constants.

Value	Meaning
DMM_OKAY	DMM initialized successfully.
Negative Value	Error code

Example

```
/* initialize DMM */
int i = DMMInit(0,"C:\SM40CAL.dat");      // Initialize the first DMM
```

DMMIsAutoRange

MODEL 1004 ☑ MODEL 1005 ☑

Description Get the status of the autorange flag.

int DMMIsAutoRange(int *nDmm*)

Remarks This function returns the DMM autorange flag state.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.

Return Value TRUE, FALSE or an error code.

Value	Meaning
TRUE	Autoranging mode is selected.
FALSE	Autoranging mode is not selected.
DMM_E_DMM	Invalid DMM number.

Example

```
int autorange = DMMIsAutoRange(0);
```

DMMIsInitialized

MODEL 1004 ☑ MODEL 1005 ☑

Description Get the status of the DMM.

int DMMIsInitialized(int *nDmm*)

Remarks This function returns the status of the DMM. If it is TRUE, then the DMM has been initialized and is active. If FALSE, the DMM is not initialized and should not be addressed. This function is used for maintenance and is not needed under normal operation.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.

Return Value TRUE, FALSE or an error code.

Value	Meaning
TRUE	DMM is initialized and active.
FALSE	DMM is not initialized.
DMM_E_DMM	Invalid DMM number.

Example

```
int active = DMMIsInitialzied(0);
```

DMMIsRelative

MODEL 1004 ☑ MODEL 1005 ☑

Description Get the status of the Relative flag.

int DMMIsRelative(**int** *nDmm*)

Remarks This function returns the DMM Relative flag state.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.

Return Value Integer TRUE, FALSE or an error code.

Value	Meaning
TRUE	Relative mode is selected.
FALSE	Relative mode is not selected.
Negative Value	Error code

Example

```
int rel = DMMIsRelative(0);
```

DMMLoadCalFile

MODEL 1004 ☑ MODEL 1005 ☑

Description Reload calibration record from file.

int DMMLoadCalFile(int *nDmm*, **LPCSTR** *lpszCal*)

Remarks This function provides the capability to reload the calibration record. This is useful in making limited calibration adjustments to the DMM. By having a copy of the original calibration file '**SM40CAL.DAT**' open with an editor, and modifying calibration entries, then reloading it using **DMMLoadCalFile**, one can instantly verify the corrections made. Make sure the '**SM40CAL.DAT**' file itself is not altered since that will void the calibration.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
lpszCal	**LPCSTR** Points to the name of the file containing the calibration constants for the DMM.

Return Value The return value is one of the following constants.

Value	Meaning
DMM_OKAY	Cal record loaded successfully.
Negative Value	Error code

Example

```
/* Load a modified copy of the original calibration file to
verify correction made to a specific entry */
int i = DMMLoadCalFile(0, "C:\CAL_A.dat");
```

DMMOpenTerminalCal

MODEL 1004 ☐ MODEL 1005 ☑

Description Calibrate the Inductance measurement function with open terminals.

int DMMOpenTerminalCal(**int** *nDmm*)

Remarks This function characterizes the Inductance measurement path and source, which is required prior to making inductance measurements. It should be performed within one hour, before using the inductance measurements. For better accuracy it should be performed more frequently. The Open Terminal calibration should be performed with the test leads open. The **DMMOpenTerminalCal()** sweeps the inductance stimulus source across the full bandwidth, and makes measurements at several points. It takes about twenty seconds to complete the process. For a complete characterization of the Inductance measurement system it is also necessary to perform the inductance zero operation with the inductance range and frequency selected, using the Relative function and with the probes shorted.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.

Return Value Integer error code.

Value	Meaning
DMM_OKAY	Operation successfully completed.
Negative Value	Error code

Example

```
int status = DMMOpenTerminalCal(0);
```

DMMPeriodStr

MODEL 1004 ☐ MODEL 1005 ☑

Description Return the next DMM period reading, formatted for printing.

int DMMPeriodStr(**int** *nDmm*, **LPSTR** *lpszReading*)

Remarks This function makes a period measurement and returns the result as a string formatted for printing. The print format is fixed to five digits plus a unit, e.g., 150.01 ms. See **DMMFrequencyStr()** for more details.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
lpszReading	**LPSTR** Points to a buffer (at least 64 characters long) to hold the converted result. The return value will consist of a leading sign, a floating-point value in exponential notation, and a unit specifier.

Return Value The return value is one of the following constants.

Value	Meaning
DMM_OKAY	Operation successfully completed.
Negative Value	Error code
DMM_CNT_RNG	Period measurement H/W is over or under range.

Example

```
char cBuf[64];
int status;
status = DMMPeriodStr(0, cBuf);
```

DMMPolledRead

MODEL 1004 ☑ MODEL 1005 ☑

Description Tests the DMM for ready status, and returns the next floating-point reading.

int DMMPolledRead(int *nDmm*, **double FAR ***_lpdResult_**)**

Remarks **DMMPolledRead()** polls the DMM for readiness. If the DMM is not ready it will return **FALSE**. If the DMM is ready with a new reading it will return **TRUE**, and the reading will be placed at the location pointed to by *lpdResult*. See **DMMPolledReadCmd()** for more details. Do not use **DMMReady()** to check for readiness since it will cause communication failure.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
lpdResult	**double FAR *** Points to a double which holds the next reading.

Return Value The return value is one of the following constants.

Value	Meaning
FALSE	DMM is not ready
TRUE	DMM is ready, and reading is placed at *lpdResult*
Negative Value	Error code

Example

```
double read;
if(DMMPolledRead(0, &d)) fprintf("%9.4f\n",d);  // Show
```

DMMPolledReadCmd

MODEL 1004 ☑ MODEL 1005 ☑

Description Send DMM Polled Read command.

int DMMPolledReadCmd(**int** *nDmm*)

Remarks If the DMM is not busy with a prior Polled read process, this function will trigger the DMM to execute a single read command. The DMM must be set to a specific range and one of the following functions to use the polled read command: VDC, VAC, IDC, IAC, 2-wire, 4-wire, 6-wire, or RTD function. Composite functions such as Capacitance, Inductance, Peak-to-Peak etc. are not capable of polled read operation. Measurement rate must be 10 rps or higher. If FALSE is returned, the DMM is busy processing a prior polled read. A DMM_OKAY indicates the DMM accepted the read command and entered the busy state. The DMM remains busy until it is ready with the next reading. This function is useful where it is necessary to conserve CPU time and make the DMM a polled device. Use **DMMPolledRead()** or **DMMPolledReadStr()** to test for readiness and read measurement. Do not use **DMMReady()** to check for readiness since it will cause communication failure.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.

Return Value DMM_OKAY if command accepted, else FALSE or an error code.

Value	Meaning
FALSE	DMM is busy and can't execute a polled read command.
DMM_OKAY	Operation successful. DMM entered busy state
Negative Value	Error code

Example

```
int status = DMMPolledReadCmd(0);
```

DMMPolledReadStr

MODEL 1004 ☑ MODEL 1005 ☑

Description If DMM is ready, return the next reading from the DMM formatted for printing.

int DMMPolledReadStr(int *nDmm*, **LPSTR** *lpszReading*)

Remarks This function is a string version of **DMMPolledRead**. See **DMMPolledRead** for more details.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
lpszReading	**LPSTR** Points to a buffer (at least 64 characters long) to hold the converted result. The return value will consist of a leading sign, a floating-point value in exponential notation, and a unit specifier.

Return Value The return value is one of the following constants, or the string length if OK.

Value	Meaning
FALSE	DMM is not ready
TRUE	DMM is ready, and reading is placed at *lpszReading*
Negative Value	Error code

Example

```
char strMsg[64];
if(DMMPolledReadStr(0, strMsg)) MessageBox(0,strMsg,  "MODEL 1005",MB_OK);
// display readings;
```

DMMRead

MODEL 1004 ☑ MODEL 1005 ☑

Description Return the next floating-point reading from the DMM.

int DMMRead(int *nDmm***, double FAR *** *lpdResult***)**

Remarks **DMMRead** reads the next result from the DMM, performs all scaling and conversion required, and returns the result as a 64-bit double-precision floating-point number in the location pointed to by *lpdResult*. It can read all the **Primary** functions (those that can be selected using **DMMSetFunction()** and **DMMSetRange()**). Returned result is a scaled value which is normilized to the selected range. That is, it returns 300 for 300mV input in the 330 mV range, and 100 for 100 kΩ input in the 330k Ω range. Use the **DMMReadNorm()** function for base units read function.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
lpdResult	**double FAR *** Points to a double that holds the next reading.

Return Value The return value is one of the following constants.

Value	Meaning
DMM_OKAY	DMM initialized successfully.
Negative Value	Error code
DMM_E_RANGE	DMM over range error occurred.

Example

```
double d;
int status;
status = DMMRead(0, &d);
```

DMMReadBuffer

MODEL 1004 ☑ MODEL 1005 ☑

Description Return the next double floating-point reading from the DMM internal buffer.

int DMMReadBuffer(int *nDmm*, **double FAR ***_lpdResult_)

Remarks Read the next measurement from the DMM's internal buffer, pointed to by an internal buffer pointer, and increment the pointer. Store the measurement as a 64-bit double-precision floating-point number in the location pointed to by *lpdResult*. See **DMMArmTrigger()** functions for more detail.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
lpdResult	**double FAR *** Points to a double that holds the reading.

Return Value The return value is one of the following constants.

Value	Meaning
DMM_OKAY	Operation successfully completed.
Negative Value	Error Code

Example

```
double Buffer[10];
      int status;
      DMMArmTrigger(0,10);    // Set up for 10 triggered samples
while( ! DMMReady(0));
      for(i=0; i < 10 ; i++)
            status = DMMReadBuffer(0, &Buffer[i]);
```

DMMReadBufferStr

MODEL 1004 ☑ MODEL 1005 ☑

Description Return the next reading, formatted for printing.

int DMMReadBufferStr(**int** *nDmm*, **LPSTR** *lpszReading*)

Remarks This function is the same as **DMMReadBuffer()** except the reading is formatted into a string with units. Measurements are stored as a null terminated string at the location pointed to by *lpszReading*.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
lpszReading	**LPSTR** Points to the location that holds the formatted reading string. Allow minimum of 64.

Return Value The return value is one of the following constants.

Value	Meaning
DMM_OKAY	Operation successfully completed.
Negative Value	Error code

Example

```
char Buf[64];
      DMMArmTrigger(0,1);      // take a single triggered sample
while( !DMMReady(0));
      DMMReadBufferStr(0, Buf);
```

DMMReadCrestFactor

MODEL 1004 ☐ MODEL 1005 ☑

Description Return ACV signal's Crest Factor.

int DMMReadCrestFactor(int *nDmm*, **double FAR ***_lpdResult_)

Remarks This is a **Secondary** function and the DMM must be in ACV measurement function, and a valid range must be set. A double-precision floating-point Crest Factor is stored in the location pointed to by *lpdResult*. This measurement is a composite function, utilizing several sub functions, and could take over 10 seconds to perform. See the Crest Factor measurement section of the manual for more detail.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
lpdResult	**double FAR *** Points to a double that holds the Crest Factor.

Return Value The return value is one of the following constants.

Value	Meaning
DMM_OKAY	Operation successfully completed.
Negative Value	Error code

Example

```
double CF; int status = DMMReadCrestFactor(0, &CF);
```

DMMReadDutyCycle

MODEL 1004 ☐ MODEL 1005 ☑

Description Return percent duty cycle of ACV signal.

int DMMReadDutyCycle(int *nDmm*, **double FAR *** *lpdDcy*)

Remarks This is a **Secondary** function and the DMM must be in AC measurement function, and a valid range must be set. It returns percent duty cycle of the signal. It is stored as double-precision floating-point number in the location pointed to by *lpdDcy*. The measured duty cycle is affected by the setting of the Threshold DAC.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
lpdDcy	**double FAR *** Points to the location which holds the duty cycle.

Return Value The return value is one of the following constants.

Value	Meaning
DMM_OKAY	Operation successfully completed.
Negative Value	Error code

Example

```
double dcy; int state; state = DMMReadDutyCycle(0, &dcy);
```

DMMReadFrequency

MODEL 1004 ☐ MODEL 1005 ☑

Description Return the next double floating-point frequency reading from the DMM.

int DMMReadFrequency(**int** *nDmm*, **double FAR ***_lpdResult_)

Remarks This function makes a single frequency measurement and stores the result as a 64-bit double-precision floating-point number in the location pointed to by *lpdResult*. For faster measurement select the frequency counter to a specific range using **DMMSetCounterRng()**. See **DMMFrequencyStr()** for more details.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
lpdResult	**double FAR *** Points to a double that holds the frequency.

Return Value The return value is one of the following constants.

Value	Meaning
DMM_OKAY	Operation successfully completed.
DMM_E_INIT	DMM is uninitialized. Must be initialized prior to using any function.
DMM_E_DMM	Invalid DMM number.
DMM_CNT_RNG	Frequency counter is over or under range.

Example

```
double d;
int status = DMMReadFrequency(0, &d);
```

DMMReadInductorQ

MODEL 1004 ☐ MODEL 1005 ☑

Description Return inductor's Q value.

int DMMReadInductorQ(**int** *nDmm*, **double FAR ***_lpdResult_)

Remarks This is a **Secondary** function and the DMM must be in the Inductance measurement function, and a valid inductance value must have been read prior to using this function. Resulting Q is stored as double-precision floating-point number in the location pointed to by *lpdResult*.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
lpdResult	**double FAR *** Points to a double that holds the inductor's Q value.

Return Value The return value is one of the following constants.

Value	Meaning
DMM_OKAY	Operation successfully completed.
Negative Value	Error code

Example

```
double Q;
int status = DMMReadInductorQ(0, &Q);
```

DMMReadMeasurement

MODEL 1004 ☑ MODEL 1005 ☑

Description Return a reading from the result of **DMMSetTrigRead** operation.

int DMMReadMeasurement(int *nDmm*, **double FAR ***lpdRead**)**

Remarks This measurement reading function is designed to read triggered measurements from the DMM. It is a fast reading function. It returns **FALSE** while no new reading is ready. If a reading is ready, TRUE is returned, and the result in the form of a 64-bit double-precision floating-point number is placed at the location pointed to by *lpdRead*. The returned value is in base units. That is, it returns 0.3 for a 300mV input and 1e6 for 1.0 Mohm measurement. This function is designed to read bursting measurements form the DMM, resulting from **DMMSetTrigRead** and **DMMBurstRead** operations.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
lpdRead	**double FAR *** Pointer to a double that holds the next reading.

Return Value Integer value version code or an error code.

Value	Meaning
TRUE	Measurement was read into **lpdRead*
FALSE	No measurement is available
TIMEOUT	Communication timeout. No reading available within 9s.
OVERRUN	Communication overrun. Controller did not keep up with DMM transmission.
Other Negative Value	Error code.

Example

```
double Reading[150];

      DMMBurstRead(0, 4, 150); // 4 settle., 150 samples
for(i=0; i < 150 ; i++)       // read 150 measurements

            while( DMMReadMeasurement(0 , Reading[i]) == FALSE ); // wait for
all measurements to be ready, and read them.
```

DMMReadMedian

MODEL 1004 ☐ MODEL 1005 ☑

Description Return ACV signal's Median value.

int DMMReadMedian(int *nDmm*, **double FAR *** *lpdResult*)

Remarks This is a **Secondary** function and the DMM must be in ACV measurement function, and a valid range must be set. A double-precision floating-point Median voltage result is stored in the location pointed to by *lpdResult*. This measurement is a composite function which utilizes several sub functions, and could take over 10 seconds to perform. See the Median measurement section of the manual for more detail.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
lpdResult	**double FAR *** Points to a double that holds the median voltage.

Return Value The return value is one of the following constants.

Value	Meaning
DMM_OKAY	Operation successfully completed.
Negative Value	Error code

Example

```
double Median; int status = DMMReadMedian(0, &Median);
```

DMMReadNorm

MODEL 1004 ☑ MODEL 1005 ☑

Description Take a reading that is in base value.

int DMMReadNorm(int *nDmm***, double FAR ****lpdRead***)**

Remarks This **Primary** read function is similar to **DMMRead()**. It returns a double floating-point reading. The returned value is corrected for base units. That is, it returns 0.3 for a 300 mV input and 1e6 for 1.0 MOhm.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
lpdRead	**double FAR *** Pointer to a location where the reading is saved.

Return Value Integer value version code or an error code.

Value	Meaning
DMM_E_RANGE	Over/Under range error.
Negative Value	Error code
DMM_OKAY	Valid return.

Example

```
double reading; int status = DMMReadNorm(0, &reading);
```

DMMReadPeakToPeak

MODEL 1004 ☐ MODEL 1005 ☑

Description Return ACV signal's peak-to-peak value.

int DMMReadPeakToPeak(int *nDmm,* **double FAR *** *lpdResult*)

Remarks This is a **Secondary** function and the DMM must be in ACV measurement function, and a valid range must be set. A double-precision floating-point peak-to-peak voltage result is stored in the location pointed to by *lpdResult*. This measurement is a composite function which utilizes several sub functions, and could take over 10 seconds to perform.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
lpdResult	**double FAR *** Points to a double that holds the Peak-to-Peak value.

Return Value The return value is one of the following constants.

Value	Meaning
DMM_OKAY	Operation successfully completed.
Negative Value	Error code

Example

```
double ptp; int status = DMMReadPeakToPeak(0, &ptp);
```

DMMReadPeriod

MODEL 1004 ☐ MODEL 1005 ☑

Description Return the next double floating-point period reading from the DMM.

int DMMReadPeriod(int *nDmm,* **double FAR *** *lpdResult*)

Remarks This is a **Secondary** function and the DMM must be in ACV measurement function, and a valid range must be set. It makes a single period measurement, and stores the result as a double-precision floating-point number in the location pointed to by *lpdResult*. See **DMMFrequencyStr()** for more details.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
lpdResult	**double FAR *** Points to a double that holds the period.

Return Value The return value is one of the following constants.

Value	Meaning
DMM_OKAY	Operation successfully completed.
Negative Value	Error code
DMM_CNT_RNG	Period measurement hardware is over or under range.

Example

```
double d;
int status;
status = DMMReadPeriod(0, &d);
```

DMMReadStr

MODEL 1004 ☑ MODEL 1005 ☑

Description Return the next reading from the DMM formatted for printing.

int DMMReadStr(int *nDmm*, **LPSTR** *lpszReading*)

Remarks This function is the string version of **DMMRead()**. It reads the next **Primary** measurement result, performs all scaling and conversion required, and returns the result as a string formatted for printing. The print format is determined by the range and function. See **DMMRead()** for more details.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
lpszReading	**LPSTR** Points to a buffer (at least 64 characters long) to hold the converted result. The return value will consist of a leading sign, a floating-point value in exponential notation, and a unit specifier.

Return Value The return value is one of the following constants, or the string length if OK.

Value	Meaning
DMM_OKAY	Valid return.
Negative Value	Error code
DMM_E_RANGE	DMM over range error occurred.

Example

```
char cBuf[64]; int status = DMMReadingStr(0, cBuf);
```

DMMReadTotalizer

MODEL 1004 ☐ MODEL 1005 ☑

Description Read the totalized value accumulated by the Totalizer function.

long DMMReadTotalizer(int *nDmm*)

Remarks This function reads the total value accumulated by the Totalizer function. For details see **DMMStartTotalize()**.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.

Return Value The return value is the totalized count, or if negative one of the following constants.

Value	Meaning
Negative Value	Error code

Example

```
long total = DMMReadTotalizer(0);
```

DMMReadWidth

MODEL 1004 ☐ MODEL 1005 ☑

Description Return the positive and negative pulse widths.

int DMMReadWidth(int *nDmm,* **double FAR *** *lpdPwid,* **double FAR *** *lpdNwid*)

Remarks This is a **Secondary** function and the DMM must be in ACV measurement function, and a valid range must be set. It returns two parameters: positive and negative pulse widths. These parameters are stored as double-precision floating-point numbers in the location pointed to by *lpdPwid and lpdNwid*. The measured widths are affected by the setting of the Threshold DAC.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
lpdPwid	**double FAR *** Points to a double that holds the positive width.
lpdNwid	**double FAR *** Points to a double that holds the negative width.

Return Value The return value is one of the following constants.

Value	Meaning
DMM_OKAY	Operation successfully completed.
Negative Value	Error code

Example

```
double pw,nw; int state; state = DMMReadWidth(0, &pw, &nw);
```

DMMReady

MODEL 1004 ☑ MODEL 1005 ☑

Description Return the ready state of the DMM following trigger operation.

int DMMReady(**int** *nDmm*)

Remarks Following the completion of a triggered measurement event, be it hardware or software, the DMM indicates the completion. The **DMMReady** function checks the DMM and returns TRUE if ready, and FALSE otherwise. Once a TRUE status is returned, the **DMMReady** function should not be used again since a TRUE also indicates that some flags have been clear, which allow further operations. See **DMMArmAnalogTrigger()**, **DMMArmTrigger()**, **DMMTrigger()**, **DMMReadBuffer(),** and **DMMPolledRead()** for more details on this function.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.

Return Value The return value is one of the following constants.

Value	Meaning
TRUE	DMM is done and buffer is ready to be read.
FALSE	DMM is not ready.
Negative Value	Error code

Example

```
double Buffer[10];                                    DMMTrigger(0,10);
while( ! DMMReady(0) );
```

for(i=0; i < 10 ; i++) j = DMMReadBuffer(0, &Buffer[i]);

DMMSetACVSource

MODEL 1004 ☐ MODEL 1005 ☑

Description Set the ACV source output level and frequency.

int DMMSetACVSource(int *nDmm***, double FAR** *ldVolts***, double FAR** *ldFreq***)**

Remarks This **Secondary** function sets the AC voltage source to RMS amplitude of *ldVolts*, and the frequency to *ldFreq*. The DMM must be in **VAC_SRC** operation for this function to execute properly. When the DMM is in **VAC_SRC** operation, and the **DMMSetACVSource()** is applied, reading the DMM **(DMMRead(), DMMReadStr()**) will return the measurement of the output voltage. This function acts on the main 12-bit source DAC. If better accuracy is needed it can be accomplished by selecting the ClosedLoop mode (**DMMSetSourceMode()**). This mode engages the Trim DAC, which augments the 12-bit DAC to produce 16 effective bits. In the ClosedLoop mode, the source level is adjusted any time the DMM is read, making small corrections until the reading is equal to ldVolts. However, for the ClosedLoop mode to update the source level, it is necessary to read the DMM multiple times. Update rate should not exceed 5 rps when using the Closed Loop mode. Two ACV measurement ranges are available in VAC_SRC mode, the 3.3 V and the 330 mV. If the Autorange mode is enabled, the DMM will automatically select the appropriate range.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
ldVolts	**double FAR** AC RMS voltage to be set. Range: 0.05 to 7.25 V RMS.
ldFreq	**double FAR** Source frequency to be set. Range: 2 Hz to 76 kHz.

Return Value Integer error code.

Value	Meaning
DMM_OKAY	Operation successfully completed.
Negative Value	Error code

Example

```
double reading; int I;                    DMMSetACVSource(0, 7.0,
1000.0); // source 7V and 1kHz DMMSetSourceMode(0, CLOSED_LOOP); // Closed
loop mode      for(I=0;I<100;I++) DMMRead(0,&reading); // update 100 times
```

DMMSetAutoRange

MODEL 1004 ☑ MODEL 1005 ☑

Description Enable/Disable autorange operation of DMM

int DMMSetAutoRange(int *nDmm*, **BOOL** *bAuto*)

Remarks This function enables or disables autorange operation of the DMM.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
bAuto	**BOOL** Determines whether or not autoranging is done. The value TRUE enables autoranging, FALSE disables it.

Return Value The return value is one of the following constants.

Value	Meaning
DMM_OKAY	Function succeeded.
Negative Value	Error code

Example

```
status = DMMSetAutoRange(0, TRUE); /* enable autoranging */
```

DMMSetBuffTrigRead

MODEL 1004 ☑ MODEL 1005 ☑

Description Setup the DMM for Triggered operation.

int DMMSetBuffTrigRead(int *nDmm***, int** *iSettle***, int** *iSamples***, int** *iEdge***)**

Remarks Setup the DMM for external hardware trigger operation. Following reception of this command the DMM enters a wait state. After reception of an external trigger edge of *iEdge* polarity, the DMM takes *iSettle* + 1 readings at the set measurement function, range, and reading rate; and stores the last reading in an internal buffer. This process is repeated for *iSamples*. This function is particularly useful in conjunction with a triggering instrument such as the SM4042 relay scanner. No autoranging is allowed in this mode. The number of trigger edges must be equal or greater than *iSamples* to properly terminate this mode. Between the time the **DMMSetBuffTrigRead()** is issued and the time the buffer is read, no other command should be sent to the DMM. One exception is the **DMMDisarmTrigger** command. This function is usable for VDC, VAC, Ohms, IAC, IDC, and RTD measurements.

Use the **DMMReady()** to monitor when the DMM is ready (following trigger(s) and the reading of *iSamples*). When ready, you can read up to *iSamples*, using **DMMReadBuffer** or **DMMReadBufferStr** functions. Once **DMMReady()** returns **TRUE**, it should not be used again prior to reading the buffer, since it prepares the buffer for reading when it detects a ready condition.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
iSettle	**int** The number of setteling measurements, prior to read value. Must be set between 0 and 120. Recommanded value is 4.
iSamples	**int** The number of samples the DMM takes following the same number of trigger pulses. This number must be between 1 and 64, inclusive.
iEdge	**int** The edge polarity of the trigger signal. 1 for Positive, or leading edge, and 0 for negative or trailing edge trigger.

Return Value The return value is one of the following constants.

Value	Meaning
DMM_OKAY	Operation successfully terminated
Negative Value	Error code.

Example

```
double Buffer[64];

      DMMSetBuffTrigRead(0, 4, 64, 0); // Negative edge, 4  //setteling
readings, and 64 samples/triggers
while( ! DMMReady(0) );       // wait for completion

      for(i=0; i < 64 ; i++)        // read buffer

            j  = DMMReadBuffer(0, &Buffer[i]);
```

DMMSetCapsMeasure

MODEL 1004 ☐ MODEL 1005 ☑

Description Tune the capacitance measurement function parameters for higher measurement speed.

int DMMSetCapsMeasure(int *nDmm,* **int** *iAverage,* **int** *iSamples*)

Remarks This function should be used carefully since it modifies the capacitance function basic measurement parameters; the averages value, *iAverage*, and the number of points sampled, *iSamples*. This function is provided only for cases where it is necessary to improve measurement speed. When using this function keep in mind that the accuracy specification provided for capacitance is not guaranteed. Also, modifying these values could have profound effect on the operation of the function. Any time a capacitance range is changed, these values are set to the default values. For instance, values of 1 and 3 for *iAverage* and *iSamples* will reduce measurement time on the 12nF range from 0.8s to about 50ms.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
iAverage	**int** The average value must be set between 1 and 100.
iSamples	**int** The number of samples must be set to at least 3.

Return Value The return value is one of the following constants.

Value	Meaning
DMM_OKAY	Valid return.
Negative Value	Error code

Example

```
double Buffer[64] DMMSetCapsMeasure(0,1,3);
```

DMMSetCompThreshold

MODEL 1004 ☐ MODEL 1005 ☑

Description Set the Threshold DAC level.

int DMMSetCompThreshold(int *nDmm***, double FAR** *ldThreshold***)**

Remarks This **Secondary** function sets the output of the Threshold DAC. To use this function, the DMM must be in AC volts. This function sets the detection threshold of the AC comparator. It is compared by the comparator to the AC coupled input voltage. This function is associated with the following functions: Totalizer, Frequency counter, Period, Pulse width, and Duty Cycle measurements. *ldThreshold* range is determined by the selected ACV range. For instance, when the 250 V AC range is selected, the allowed range of *ldThreshold* is –500 V to +500 V. See the specification section for more details.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
ldThreshold	**double FAR** DC voltage to be set. Allowed range depends on selected ACV range.

Return Value Integer error code.

Value	Meaning
DMM_OKAY	Operation successfully completed.
Negative Value	Error code

Example

```
DMMSetCompThreshold(0,28.5); // Set comp. threshold to 28.5V
```

DMMSetCounterRng

MODEL 1004 ☐ MODEL 1005 ☑

Description Set the frequency counter to a specific range.

int DMMSetCounterRng(int *nDmm***, int** *fRange***)**

Remarks This function forces the auto-ranging frequency counter to a specific range, *fRange*. Use this function if the approximate frequency to be measured is known. It will eliminate the time necessary for the counter to autorange to the appropriate range. It saves time by removing the requirement to make multiple frequency measurements in order to allow the counter to range. All ranges are defind in ***UserDMM.h*** file.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
fRange	**int** The range to be set is a value between 0 and 7. See ***UserDMM.h.***

Return Value Integer error code.

Value	Meaning
DMM_OKAY	Operation successfully completed.
Negative Value	Error code

Example

```
DMMSetCounterRng(0, COUNTR_320HZ); // Set counter to measure a
frequency between 65Hz to 320Hz
```

DMMSetDCISource

MODEL 1004 ☐ MODEL 1005 ☑

Description Set the DCI source output level.

int DMMSetDCISource(int *nDmm***, double FAR** *ldAmps***)**

Remarks This **Secondary** function sets the DC current source to *ldAmps*. The DMM must be in **IDC_SRC** for this function to execute properly. Further, the appropriate DCI range must be selected. When the DMM is in **IDC_SRC** operation, and the **DMMSetDCISource()** is applied, reading the DMM (**DMMRead()** or **DMMReadStr()**) will return the output voltage measurement. This function acts on the main 12-bit source DAC. If better resolution is needed it can be accomplished by setting the Trim DAC by using the **DMMSetTrimDAC** function. There are five current source ranges. The DMM reads the output (load) voltage using the 33 V range.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
ldAmps	**double FAR** DC current to be set. Can be 0 to 1.25 X selected range.

Return Value Integer error code.

Value	Meaning
DMM_OKAY	Operation successfully completed.
Negative Value	Error code

Example

```
DMMSetRange(0, _1uA)          // Select 1uA source range
DMMSetDCISource(0, 1.1e-6);   // Set source to 1.1uA
```

DMMSetDCVSource

MODEL 1004 ☐ MODEL 1005 ☑

Description Set the DCV source output level.

int DMMSetDCVSource(int *nDmm***, double FAR** *ldVolts***)**

Remarks This **Secondary** function sets the DC voltage source to *ldVolts*. The DMM must be in **VDC_SRC** for this function to execute properly. When the DMM is in **VDC_SRC** operation, and the **DMMSetDCVSource()** is applied, reading the DMM (**DMMRead()** or **DMMReadStr()**) will return the measurement of the output voltage. This function acts on the main 12-bit source DAC. If better accuracy is needed it can be accomplished by selecting the ClosedLoop mode (**DMMSetSourceMode()**). This mode engages the Trim DAC, which augments the 12-bit DAC to produce 16 effective bits. In ClosedLoop mode, the source level is adjusted every time the DMM is read, making small corrections until the reading is equal to *ldVolts*. However, for the ClosedLoop mode to update the source level, it is necessary to read the DMM multiple times. Update rate should not exceed 10 rps when using the Closed Loop mode. The DMM reads voltages using the 33 V range.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
ldVolts	**double FAR** DC voltage to be set. Can be –10.5 to 10.5 V.

Return Value Integer error code.

Value	Meaning
DMM_OKAY	Operation successfully completed.
Negative Value	Error code

Example

```
double reading; int I;   DMMSetDCVSource(0, 1.25); // Set source
to 1.25V DMMSetSourceMode(0, CLOSED_LOOP); // Closed loop mode
for(I=0;I<100;I++) DMMRead(0,&reading); // update 100 times
```

DMMSetFuncRange

MODEL 1004 ☑ MODEL 1005 ☑

Description Set the DMM function and range.

int DMMSetFuncRange(**int** *nDmm*, **int** *nFuncRnge*)

Remarks This function sets both, the function and the range used by the DMM. The table of values is defined as VDC_330mV, VAC_3300mV, IDC_330mA, OHM_4W_330K, etc. definitions in the ***UserDMM.h*** header file.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
nFuncRnge	**int** A pre-defined constant corresponding to the desired function and range.

Return Value The return value is one of the following constants.

Value	Meaning
DMM_OKAY	Operation successfully completed.
Negative Value	Error code
DMM_E_FUNC	Invalid DMM function.

Example

```
status = DMMSetFuncRange(0, VDC_33V);
```

DMMSetFunction

MODEL 1004 ☑ MODEL 1005 ☑

Description Set the DMM function.

int DMMSetFunction(int *nDmm*, **int** *nFunc*)

Remarks This function sets the function used by the DMM. The table of values is defined by the *VDC*, *VAC*, *IDC*, *IAC*, *OHMS2W*, and *OHMS4W*, *etc* definitions in the DMM header file. Not all functions are available for all DMM types. For instance, the MODEL 1005 has Capacitance while the MODEL 1004 does not.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
nFunc	**int** A pre-defined constant corresponding to the desired function.

Return Value The return value is one of the following constants.

Value	Meaning
DMM_OKAY	Operation successfully completed.
Negative Value	Error code
DMM_E_FUNC	Invalid DMM function.

Example

```
status = DMMSetFunction(0, INDUCTANCE);
```

DMMSetInductFreq

MODEL 1004 ☐ MODEL 1005 ☑

Description Set the frequency of the Inductance Source.

int DMMSetInductFreq(int *nDmm***, double FAR** *lpdFreq***)**

Remarks This function sets the frequency of the Inductance measurement source. The value of the frequency should be between 20 Hz and 75 kHz. This function overrides the default frequency for each of the inductance ranges. Therefore, setting a new Inductance measurement range changes the frequency. Use this function after setting the range.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
lpdFreq	**double FAR** Frequency to be set.

Return Value Integer error code.

Value	Meaning
DMM_OKAY	Operation successfully completed.
Negative Value	Error code

Example

```
int status = DMMSetInductFreq(0, 10e3); // Set source to 10kHz
```

DMMSetRange

MODEL 1004 ☑ MODEL 1005 ☑

Description Set the DMM range for the present function.

int DMMSetRange(**int** *nDmm*, **int** *nRange*)

Remarks This function sets the range used by the DMM for the present function. The table of values is defined by the *_330mV, _3300uA, etc.* definitions in the DLL header file. Not all ranges are available for all DMM types. For instance the MODEL 1005 has a 33 Ohms range, and the MODEL 1004 does not.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
nRange	**int** A pre-defined constant corresponding to the desired range.

Return Value The return value is one of the following constants.

Value	Meaning
DMM_OKAY	Operation successfully completed.
Negative Value	Error code
DMM_E_RANGE	Invalid DMM range value.

Example

```
status = DMMSetRange(0, _330mA);
```

DMMSetRate

MODEL 1004 ☑ MODEL 1005 ☑

Description Set the DMM reading rate.

int DMMSetRate(int *nDmm,* **int** *nRate*)

Remarks This function sets the reading rate used by the DMM. The table of values is defined by the RATE_ values in the header file.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
nRate	**int** A pre-defined constant (RATE_*) corresponding to the desired reading rate.

Return Value The return value is one of the following constants.

Value	Meaning
DMM_OKAY	Operation successfully completed.
Negative Value	Error code
DMM_E_RATE	Invalid DMM reading rate.

Example

```
status = DMMSetRate(0, RATE_0P1);   // Set to 0.1rps
```

DMMSetRelative

MODEL 1004 ☑ MODEL 1005 ☑

Description Set the DMM relative reading mode for the present function.

int DMMSetRelative(int *nDmm*, **BOOL** *bRelative*)

Remarks This function selects relative or absolute reading mode for the DMM. If the *bRelative* parameter value is TRUE, the DMM will change to relative reading mode. If FALSE, the DMM will change to absolute reading mode. Caution: Do not select **DMMSetRelative()** when in the autorange mode.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
bRelative	**BOOL** TRUE to enter relative mode, FALSE to clear mode.

Return Value The return value is one of the following constants.

Value	Meaning
DMM_OKAY	DMM mode changed successfully.
Negative Value	Error code

Example

```
status = DMMSetRelative(0, TRUE);
```

DMMSetRTD

MODEL 1004 ☐ MODEL 1005 ☑

Description Set the RTD parameters.

int DMMSetRTD(int *nDmm***, int** *iWires***, double FAR** *ldRo***)**

Remarks This **Secondary** function sets the RTD parameters. The DMM must be in **RTD** measurement function for this function to execute properly. *iWires* selects between 3-wire and 4-wire RTD (3-wire RTDs are not implemented in this version of software). *ldRo* sets the RTD R_o (Ice point resistance). This function must follow the selection of the basic RTD type, using **DMMSetRange()**, since it modifies the default R_o parameter for the selected RTD.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
iWires	**int** RTD's number of connecting wires RTD_4_W or RTD_3_W
ldRo	**double FAR** R_o resistance. See specs for allowed range for each RTD type.

Return Value Integer error code.

Value	Meaning
DMM_OKAY	Operation successfully completed.
Negative Value	Error code

Example

```
DMMSetFunction(0, RTD);         // RTD measurement function
DMMSetRange(0, _pt385);         // Select RTD
DMMSetRTD(0, RTD_4_W, 1000.0); // Set Ro = 1k Ohms
```

DMMSetSourceMode

MODEL 1004 ☐ MODEL 1005 ☑

Description Set the DCV and ACV sources to ClosedLoop or OpenLoop mode.

int DMMSetSourceMode(int *nDmm***, int** *iMode***)**

Remarks This **Secondary** function sets the DC and AC voltage sources to either **OPEN_LOOP** or **CLOSED_LOOP**. In **CLOSED_LOOP** the sources use the main 12-bit source DAC. In **CLOSED_LOOP** the Trim DAC is also used, which augments the 12-bit DAC to produce 16 effective bits. Open loop updates are very quick. In ClosedLoop mode the source level is adjusted every time the DMM is read, making small corrections until the reading is equal to the set voltage. However, for the ClosedLoop mode to update the source level, it is necessary to read the DMM multiple times. See **DMMSetDCVSource()** and **DMMSetACVSource()** for more details.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
iMode	**int** Source adjustment mode: CLOSED_LOOP or OPEN_LOOP.

Return Value Integer error code.

Value	Meaning
DMM_OKAY	Operation successfully completed.
Negative Value	Error code

Example

```
DMMSetSourceMode(0, CLOSED_LOOP); // Select closed loop mode
```

DMMSetSynchronized

MODEL 1004 ☑ MODEL 1005 ☑

Description Enable or disable Synchroneous operation of the DMM.

int DMMSetSynchronized(int *nDmm*, **BOOL** *bSync*)

Remarks This function enables or disables the Synchronized operation of the DMM. The default operation is non-synchronized. Select the Synchronized mode when it is necessary to settle full scale input transitions from one reading to the next, and maintain the accuracy of the DMM. This is appropriate for VDC, Ohms, Leakage, DCI, Diode, and Guarded Ohms. The result of the synchronized mode is a reduced measurement rate. To run synchronized, reading rate must be set to 10 rps or higher.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
bSync	**BOOL** Determines whether or not synchronized operation is enabled. TRUE enables and FALSE disables synchronization. The default is FALSE.

Return Value The return value is one of the following constants.

Value	Meaning
DMM_OKAY	Function succeeded.
Negative Value	Error code

Example

```
int status = DMMSetSynchronized(0, FALSE); // Cancell sync.
```

DMMSetTempUnits

MODEL 1004 ☐ MODEL 1005 ☑

Description Set temperature units to °C or °F.

int DMMSetTempUnits(int *nDmm*, **int** *iTempUnits*)

Remarks This function sets the temperature units to either °C or °F. This is applicable to both the on-board temperature sensor and the RTD measurements.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
iTempUnits	**int** Temperature units can be either DEG_F for °F, or DEG_C for °C. The default is °C.

Return Value The return value is one of the following constants.

Value	Meaning
DMM_OKAY	Function succeeded.
Negative Value	Error code

Example

```
int status = DMMSetTempUnits(0, DEG_F) // set units to °F
```

DMMSetTrigRead

MODEL 1004 ☑ MODEL 1005 ☑

Description Setup the DMM for mutiple Triggered readings operation.

int DMMSetTrigRead(int *nDmm***, int** *iSettle***, int** *iSamples***, int** *iEdge***)**

Remarks Setup the DMM for external hardware trigger operation. Following reception of this command the DMM enters a wait state. After reception of an external trigger edge of *iEdge* polarity, the DMM takes *iSettle* + 1 readings at the set measurement function, range, and reading rate; and sends the last reading. This process is repeated for *iSamples* times. *iSamples* Trigger pulses must be issued to complete this process. No autoranging is allowed in this mode. The number of trigger edges must be equal or greater than *iSamples* to properly terminate this mode. Following the issue of the **DMMSetTrigRead** command, and until the sampling process ends, it is necessary to read the samples from the DMM using the **DMMReadMeasurement** command. This will prevent an Overrun communication error. In other words, the rate at which measurement are read must keep up with the DMM transmission of readings. The DMM has a built-in 5 readings FIFO to help with this problem. This function is usable for VDC, VAC, Ohms, IAC, IDC, and RTD measurements. Use the **DMMReadMeasurement()** to monitor for data avialability, and to read this data.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
iSettle	**int** The number of setteling measurements, prior to read value. Must be set between 0 and 120. Recommended value is 4.
iSamples	**int** The number of samples the DMM takes following the same number of trigger pulses. This number must be between 1 and 250, inclusive.
iEdge	**int** The edge polarity of the trigger signal. 1 for Positive, or leading edge, and 0 for negative or trailing edge trigger.

Return Value The return value is one of the following constants.

Value	Meaning
DMM_OKAY	Operation successfully terminated
Negative Value	Error code.

Example

```
double Reading[150];

      DMMSetTrigRead(0, 4, 150, 0); // Negative edge, 4  //setteling
readings, and 150 samples/triggers
for(i=0; i < 150 ; i++)       // read buffer

            while( ! DMMReadMeasurement(0 , Reading[i]) );
```

DMMSetTrimDAC

MODEL 1004 ☐ MODEL 1005 ☑

Description Set the Trim DAC level.

int DMMSetTrimDAC(int *nDmm***, int** *iValue***)**

Remarks This **Secondary** function sets the Trim DAC to a value between 0 and 100. The trim DAC can be set to augment the main 12-bit DAC, whenever it is not automatically performed, such as in VDC and VAC source while **OPEN_LOOP** mode is selected. An example would be in DCI source, or when setting the Comparator Threshold. This function consumes a lot of the on-board microcontroller's resources and must be turned off when not in use. Use **DMMDisableTrimDAC()** to turn off. With the Trim DAC the effective resolution of the composite DAC is increased to 16 bits. With *iValue* set to 100, the Trim DAC adds slightly over 1 LSB of the 12-bit DAC. See **DMMSetDCVSource()** and **DMMSetACVSource()** for more details.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
iValue	**int** Amplitude can be set from 0 to 100, corresponding to 0% to 100% Trim DAC level.

Return Value Integer error code.

Value	Meaning
DMM_OKAY	Operation successfully completed.
Negative Value	Error code

Example

```
DMMSetDCVSource(0, 5.0); // Set source to 5V
DMMSetTrimDAC(0, 50);    // add about 2.5mV to output
```

DMMStartTotalizer

MODEL 1004 ☐ MODEL 1005 ☑

Description Clear the totalized value and start the totalizer.

int DMMStartTotalizer(**int** *nDmm*, **int** *Edge*)

Remarks This is a **Secondary** function and the DMM must be in ACV measurement function, and a valid range must be selected. This function clears the Totalized count, sets the edge sense, and starts the Totalizer. The totalized value can be read during the accumulation period. However, it could affect the count by the interruption. If no reads are performed during accumulation, the input rate can be as high as 45 kHz. If reads are performed during the accumulation period, this rate could be as low as 20 kHz. The Threshold DAC sets the level at which the signals are counted. During accumulation, no other command (except **DMMReadTotalizer()**) should be used. When done, this function must be turned off using **DMMStopTotalizer()**. After the Totalizer is stopped, the accumulated result can be read using **DMMReadTotalizer**. A normal procedure would be to set the DMM to the ACV function, select voltage range, set the Threshold DAC, start the totalizer, wait for the time required, stop and read the total. The total number of events is limited to 1,000,000,000. The MODEL 1005 allows up to 90 kHz input, but reduces the resolution of the count.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
Edge	**int** Identifies the edge of the counter. If TRAILING (0) count negative edges, if LEADING (1) count positive edges.

Return Value Integer error code.

Value	Meaning
DMM_OKAY	Operation successfully completed.
Negative Value	Error code

Example

```
int status = DMMStartTotalizer(0, LEADING);
```

DMMStopTotalizer

MODEL 1004 ☐ MODEL 1005 ☑

Description Terminate the accumulation process of the Totalizer.

int DMMStopTotalizer(int *nDmm*)

Remarks This function stops the accumulation process. Following this function, the totalized value can be read. For details see **DMMStartTotalizer()**.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.

Return Value The return value is one of the following constants.

Value	Meaning
DMM_OKAY	Operation was successful.
Negative Value	Error code

Example

```
int status = DMMStopTotalizer(0);
```

DMMTerminate

MODEL 1004 ☑ MODEL 1005 ☑

Description Terminate DMM operation (DLL)

int DMMTerminate(int *nDmm*)

Remarks Removes DMM number *nDmm*. This routine is used only where it is needed to terminate one DMM and start a new one at the same *nDmm* location. Otherwise, it is not recommended to use this function.

Parameter	Type/Description
nDmm	**int** Identifies the DMM to be suspended.

Return Value The return value is one of the following constants.

Value	Meaning
TRUE	DMM Terminated
FALSE	DMM was not initialized, termination is redundant.

Example

```
DMMTerminate(0); /* Terminate DMM # 0 */
```

DMMTrigger

MODEL 1004 ☑ MODEL 1005 ☑

Description Software Trigger the DMM. Take *iSamples*.

int DMMTrigger(**int** *nDmm*, **int** *iSamples*)

Remarks Following reception of this command, the DMM makes *iSamples* readings at the currently set function, range and rate, and stores them in an internal buffer. Rate can be set between 10 to 1000 readings per second. No autoranging is allowed for this trigger operation. Between the time the **DMMTrigger** command is issued and the time the buffer is read, no other commands should be sent to the DMM. Use the **DMMReady** function to monitor when the DMM is ready (ready implies completion of *iSamples*). When ready, you can optionally read a single reading or up to *iSamples*, using **DMMReadBuffer()**.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
iSamples	**int** The number of samples the DMM takes following a trigger pulse. This number must be between 1 and 64, inclusive.

Return Value The return value is one of the following constants.

Value	Meaning
DMM_OKAY	Operation successfully terminated.
DMM_E_INIT	DMM is uninitialized. Must be initialized prior to using any function.
DMM_TRIG_N	Measurement count is out of allowed range.
DMM_E_DMM	Invalid DMM number.

Example

```
double Buffer[64];
      int state;
      DMMTrigger(0,64);
while( ! DMMReady(0));
            for(i=0; i < 64 ; i++)
            state = DMMReadBuffer(0, &Buffer[i]);
```

DMMWidthStr

MODEL 1004 ☐ MODEL 1005 ☑

Description Return positive and negative pulse width in string format.

int DMMWidthStr(int *nDmm*, **LPSTR** *lpszPos*, **LPSTR** *lpszNeg*)

Remarks This **Secondary** function is the string equivalent of **DMMReadWidth()**. The measurement results are stored at the location pointed to by *lpszPos and lpszNeg*. See **DMMReadWidth()** for more details.

Parameter	Type/Description
nDmm	**int** Identifies the DMM. DMMs are numbered starting with zero.
lpszPos	**LPSTR** Points to a buffer (at least 64 characters long) to hold the positive width result.
lpszNeg	**LPSTR** Points to a buffer (at least 64 characters long) to hold the negative width result.

Return Value The return value is one of the following constants.

Value	Meaning
DMM_OKAY	Valid return.
Negative Value	Error code

Example

```
char P[64],N[64]; int status = DMMWidthStr(0,P,N);
```

6.0 Maintenance

Warning

These service instructions are for use by qualified personnel only. To avoid electric shock, do not perform any procedures in this section unless you are qualified to do so.

This section presents maintenance information for the DMM.

Test equipment recommended for calibration is listed below. If the recommended equipment is not available, equipment that meets the indicated minimum specifications may be substituted. In general, the calibration equipment should be at least three times more accurate than the DMM specifications.

Table 9-1. Recommended Test Equipment

Instrument Type	Minimum Specifications	Recommended Model
Multi-Function Calibrator	DC Voltage Range: 0-300 V Voltage Accuracy: 9 ppm AC Voltage Range: 0-250 V Voltage Accuracy: 0.014% Resistance Range: 0-330 MΩ Resistance Accuracy: 22 ppm DC Current Range: 0-2.5 A Current Accuracy: 0.008% AC Current Range: 50 uA – 2.5 A Current Accuracy: 0.05% Capacitance Range: 10 ηF – 10 mF Capacitance Accuracy: 0.19%	Fluke 5520A

6.1 Performance Tests

This test compares the performance of the MODEL 1004 with the specifications given in Section 2. The test is recommended as an acceptance test when the instrument is first received, and as a verification after performing the calibration procedure. To ensure proper performance, the test must be performed with the MODEL 1004 installed in a personal computer, with the covers on. The ambient temperature must be between 18°C to 28°C. Allow the MODEL 1004 to warm up at least one-half hour before performing any of the tests. The default reading rate of the MODEL 1004 should be used in each test.

6.2 DC Voltage Test

The following procedure may be used to verify the accuracy of the DCV function:

1. If you have not done so, install the MODEL 1004 and place the covers back on to the computer. Ensure that the computer has been on for at least one-half hour, with the covers on, before conducting this test.

2. Apply a high quality copper wire short to the MODEL 1004 **V,Ω + & -** inputs. Select the DCV function, Autorange. Allow the MODEL 1004 to settle for several seconds, and perform the **Relative** function.

3. Apply the following DC voltages to the **V, Ω + & -** terminals. Check to see that the displayed reading on the MODEL 1004 is within the indicated range.

Table 9-2. DC Voltage Test

Step	Range	Input	Minimum Reading	Maximum Reading
1	330 mV	0V (short)	-8 μV	+8 μV
2	330 mV	190 mV	189.9787 mV	190.0213 mV
3	330 mV	-190 mV	-190.0213 mV	–189.9787 mV
4	3.3 V	1.9 V	1.899898 V	1.900103 V
5	3.3 V	-1.9 V	-1.900103 V	-1.899898 V
6	33 V	19 V	18.99834 V	19.00166 V
7	33 V	-19 V	-19.00166 V	-18.99834 V
8	330 V	190 V	189.9833 V	190.0167 V
9	330 V	-190 V	-190.0167 V	-189.9833 V

6.3 Resistance Test, 2-wire

The following procedure may be used to verify the accuracy of the 2-wire function.

1. If you have not done so, install the MODEL 1004 and place the covers back on to the computer. Ensure that the computer has been on for at least one-half hour, with the covers on, before conducting this test.

2. Connect the MODEL 1004 **V,Ω** + **&** - terminals to the calibrator HI & LO Outputs. Output

0 Ω from the calibrator. Allow the MODEL 1004 to settle for a few seconds, and perform the **Relative** function. (This effectively nulls out the lead resistance of your cabling. If you are using a Fluke 5700A or 5520A Calibrator, the 2-wire Compensation feature will give a more accurate 2-wire ohms measurement. See the *Fluke Operator's Manual* for further instructions.)

3. Apply the following Resistance values to the **V, Ω + & -** terminals . Check to see that the displayed reading on the MODEL 1004 is within the indicated range.

Table 9-3 Resistance Test, 2-wire

Step	Range	Input	Minimum Reading	Maximum Reading
1	33 Ω [1]	10 Ω	9.9972 Ω	10.0028 Ω
2	330 Ω	100 Ω	99.987 Ω	100.013 Ω
3	3.3 kΩ	1 kΩ	0.999917 kΩ	1.000083 kΩ
4	33 kΩ	10 kΩ	9.99905 kΩ	10.00095 kΩ
5	330 kΩ	100 kΩ	99.986 kΩ	100.014 kΩ
6	3.3 MΩ	1 MΩ	0.99942 MΩ	1.00058 MΩ
7	33 MΩ	10 MΩ	9.973 MΩ	10.027 MΩ
8	330 MΩ [1]	100 MΩ	97.92 MΩ	102.08 MΩ

[1] MODEL 1005 only

6.4 Resistance Test, 4-wire

The following procedure may be used to verify the accuracy of the 4-wire function.

1. If you have not done so, install the MODEL 1004 and place the covers back on to the computer. Ensure that the computer has been on for at least one-half hour, with the covers on, before conducting this test.

2. Connect the MODEL 1004 **V,Ω + & -** terminals to the calibrator HI & LO Output. Connect the MODEL 1004 **I, 4WΩ + & -** terminals to the HI & LO Sense terminals.

3. Select the 4WΩ function on the MODEL 1004, Autorange. Set the calibrator to 0 Ω. Be certain that the calibrator is set to external sense ("EX SNS" on the Fluke 5700A or "4-Wire Comp" on the 5520A). Allow the MODEL 1004 to settle for a few seconds, and perform the **Relative** function.

4. Apply the following Resistance values to the **V, Ω + & -** terminals. Check to see that the displayed reading on the MODEL 1004 is within the indicated range.

Table 9-4 Resistance Test, 4-wire

Step	Range	Input	Minimum Reading	Maximum Reading
1	33 Ω [1]	0 Ω	-2 mΩ	2 mΩ
1	33 Ω [1]	10 Ω	9.9972 Ω	10.0028 Ω
1	330 Ω	0 Ω	-6 mΩ	6 mΩ
2	330 Ω	100 Ω	99.987 Ω	100.013 Ω
3	3.3 kΩ	0 Ω	-33 mΩ	33 mΩ
4	3.3 kΩ	1 kΩ	0.999917 kΩ	1.000083 kΩ
5	33 kΩ	0 Ω	-350 mΩ	350 mΩ
5	33 kΩ	10 kΩ	9.99905 kΩ	10.00095 kΩ
5	330 kΩ	0 Ω	-5 Ω	5 Ω
6	330 kΩ	100 kΩ	99.986 kΩ	100.014 kΩ

[1] MODEL 1005 only.

Note: The use of 4-wire Ohms for resistance values above 300 kΩ is not recommended.

6.5 AC Voltage Test

The following procedure may be used to verify the accuracy of the ACV function:

1. If you have not done so, install the MODEL 1004 and place the covers back on to the computer. Ensure that the computer has been on for at least one-half hour, with the covers on, before conducting this test.

2. Apply the following AC voltages to the **V, Ω + & -** terminals. Check to see that the displayed reading on the SMX2040 is within the indicated readings range.

Table 9-5. Mid-Frequency AC Voltage Tests

All inputs are a sine wave at **400 Hz**.

Step	Range	Input	Minimum Reading	Maximum reading
1	330 mV	10 mV	9.8650 mV	10.1350 mV
2	330 mV	190 mV	189.5950 mV	190.4050 mV
4	3.3 V	100 mV	0.098735 V	0.101265 V
5	3.3 V	1.9 V	1.897565 V	1.902435 V
6	33 V	1 V	0.98327 V	1.01673 V
7	33 V	19 V	18.97313 V	19.02687 V
8	250 V	10 V	9.864 V	10.136 V
9	250 V	190 V	189.756 V	190.244 V

Table 9-6. High-Frequency AC Voltage Tests

All inputs are at **50 kHz**.

Step	Range	Input	Minimum Reading	Maximum Reading
1	330 mV	10 mV	9.707 mV	10.293 mV
2	330 mV	190 mV	188.573 mV	191.427 mV
4	3.3 V	100 mV	0.0978 V	0.1022 V
5	3.3 V	1.9 V	1.8852 V	1.9148 V
6	33 V	1 V	0.9715 V	1.0285 V

7	33 V	19 V	18.9085 V	19.0915 V
8	250 V	10 V	9.755 V	10.245 V
9	250 V	100 V	99.35 V	100.65 V

6.6 DC Current Test

The following procedure may be used to verify the accuracy of the DCI function:

1. If you have not done so, install the MODEL 1004 and place the covers back on to the computer. Ensure that the computer has been on for at least one-half hour, with the covers on, before conducting this test.

2. Remove all connections from the MODEL 1004 inputs. Select the DCI function, Autorange. Allow the MODEL 1004 to settle for a few seconds, and perform the **Relative** function.

3. Apply the following DC currents to the **I,4Ω + & -** terminals. Check to see that the displayed reading on the SMX2040 is within the indicated readings range.

Table 9-7. DC Current Test

Step	Range	Input	Minimum Reading	Maximum reading
1	3.3 mA	0 mA (open)	-0.0004 mA	0.0004 mA
2	3.3 mA	1 mA	0.9986 mA	1.0014 mA
3	33 mA	0 mA (open)	-0.003 mA	0.003 mA
4	33 mA	10 mA	9.987 mA	10.013 mA
5	330 mA	0 mA (open)	-0.060 mA	0.060 mA
6	330 mA	100 mA	99.865 mA	100.135 mA
7	2.5 A	0 A	-0.00035 A	0.00035 A
8	2.5 A	1 A	0.99315 A	1.00685 A

6.7 AC Current Test

The following procedure may be used to verify the accuracy of the ACI function:

1. If you have not done so, install the MODEL 1004 and place the covers back on to the computer. Ensure that the computer has been on for at least one-half hour, with the covers on, before conducting this test.

2. Remove all connections from the MODEL 1004 inputs. Select the ACI function, Autorange.

3. Apply the following AC currents to the **I,4Ω + & -** terminals. Check to see that the displayed reading on the SMX2040 is within the indicated readings range.

Table 9-8. AC Current Test

All Inputs are at **400Hz**

Step	Range	Input	Minimum Reading	Maximum reading
1	3.3 mA	0.1 mA	0.09588 mA	0.100412 mA
2	3.3 mA	1 mA	0.9948 mA	1.0052 mA
3	33 mA	1 mA	0.9684 mA	1.0316 mA
4	33 mA	10 mA	9.954 mA	10.046 mA
5	330 mA	10 mA	9.758 mA	10.242 mA
6	330 mA	100 mA	99.56 mA	100.44 mA
7	2.5 A	100 mA	0.09535 A	0.10465 A
8	2.5 A	1 A	0.9895 A	1.0105 A

6.8 Capacitance Test (MODEL 1005 only)

The following procedure may be used to verify the accuracy of the Capacitance function.

1. If you have not done so, install the DMM and place the covers back on to the computer. Ensure that the computer has been on for at least one-half hour, with the covers on, before conducting this test.

2. Connect the DMM **V,Ω** + **& -** terminals to the calibrator HI & LO Outputs. Attach the test leads to the DMM, leaving the other end open circuited. Allow the DMM to settle for a few seconds, and perform the **Relative** function. (This effectively nulls out the lead capacitance of your cabling.

3. Apply the following Capacitance values to the **V, Ω + & -** terminals. Check to see that the displayed reading on the /44 is within the indicated range of readings.

Step	Range	Input	Minimum Reading	Maximum reading
1	10 ηF	10 ηF	9.785 ηF	10.215 ηF
2	100 ηF	100 ηF	99 ηF	101 ηF
3	1 μF	1 μF	0.99 μF	1.01 μF
4	10 μF	10 μF	9.9 μF	10.1 μF
5	100 μF	100 μF	99 μF	101 μF
6	1 mF	1 mF	0.988 mF	1.012 mF
7	10 mF	10 mF	9.8 mF	10.2 mF

6.9 Frequency Counter Test (MODEL 1005 only)

The following procedure may be used to verify the accuracy of the Frequency Counter:

1. If you have not done so, install the DMM and place the covers back on to the computer. Ensure that the computer has been on for at least one-half hour, with the covers on, before conducting this test.

2. Select the ACV function, autorange. Turn **freq** on.

3. Apply the following AC voltages to the **V, Ω + & -** terminals. Check to see that the displayed reading on the /44 is within the indicated range of readings.

Table 9-9. ACV Frequency Counter Test

Step	Range	Input	Minimum Reading	Maximum reading
1	330 mV	33 mV, 40 Hz	39.9952 Hz	40.0048 Hz
2	3.3 V	330 mV, 40 Hz	39.9952 Hz	40.0048 Hz
3	33 V	3.3 V, 40 Hz	39.9952 Hz	40.0048 Hz
4	330 V	33 V, 40 Hz	39.9952 Hz	40.0048 Hz
5	330 mV	250 mV, 100 kHz	99.996 kHz	100.004 kHz
6	33 V	25 V, 100 kHz	99.996 kHz	100.004 kHz

2. Select the ACI function, autorange. Turn **freq** on.

3. Apply the following AC currents to the **I,4Ω + & -** terminals. Check to see that the displayed reading on the MODEL 1004 is within the tolerance appropriate for your application (e.g. 90 day or 1 year accuracy).

Table 9-10. ACI Frequency Counter Test

Step	Range	Input	Counter Reading	Tolerance
1	3.3 mA	330 uA, 40 Hz		

2	33 mA	15 mA, 40 Hz		
3	330 mA	150 mA, 40 Hz		

6.10 Calibration

Each MODEL 1004 DMM uses its own **SM40CAL.DAT** calibration file to ensure the accuracy of its functions and ranges. The **SM40CAL.DAT** file is a text file that contains the DMM identification number, calibration date, and calibration constants for all DMM ranges. For most functions, the calibration constants are scale factor and offset terms that solve the "y = mx + b" equation for each range An input "x" is corrected using a scale factor term "m" and an offset term "b"; this gives the desired DMM reading, "y". Keep in mind that for ranges and functions that are unavailable for a particular product in the MODEL 1004 family, the calibration record contains a place-holder. An example **SM40CAL.DAT** is shown:

```
card_id  10123    type 2044 calibration_date 06/15/1999
ad       ; A/D compensation
72.0     20.0
vdc      ; VDC 330mV, 3.3V, 33V, 330V ranges, offset and gain parameters
-386.0   0.99961
-37.0    0.999991
-83.0    0.999795
-8.8     1.00015
vac      ; VAC 1st line - DC offset. Than offset, gain and freq each range 330mV to 250V
5.303
0.84     1.015461          23
0.0043   1.0256            23
0.0      1.02205           0
0.0      1.031386          0
idc      ; IDC 3.3mA to 2.5A ranges, offset and gain
-1450.0  1.00103
-176.0   1.00602
-1450.0  1.00482
-176.0   1.00001
iac      ; IAC 3.3mA to 2.5A ranges, offset and gain
1.6      1.02402
0.0      1.03357
1.69     1.00513
0.0      1.0142
2w-ohm   ; Ohms 33, 330, 3.3k,33k,330k,3.3M,33M,330Meg ranges, offset and gain
1.27e+4  1.002259
1256.0   1.002307
110.0    1.002665
0.0      1.006304
0.0      1.003066
```

0.0 1.001848

0.0 0.995664

0.0 1.00030

The first column under any function, e.g.,"vdc", is the offset term "b", expressed as a value proportional to analog-to-digital (a/d) counts. The second column is the scale factor term "m". Within each function, the "b" and "m" terms are listed with the lowest range at the beginning. For example, under "2w-ohm" above, "1.27e+4 1.002259" represents the offset term for the 33 Ω range, and "1.002259" is the scale factor for this range. This record must be for the MODEL 1005 since the MODEL 1004 does not have the 33 Ohms range, and therefore these values will be set to 0.0 and 1.0.

For the ACV function, the first line in the calibration record is the DC offset value. The rest of the lines contain the RMS offset, gain correction factor, and a third column that represents a digital code from 0 to 31 that controls the high frequency performance of each AC function. A large value, e.g., 31, implies high attenuation.

The **SM40CAL.DAT** file is created by performing external calibration. The general calibration algorithm consists of applying a zero value to the DMM followed by a value of 2/3rd of the top of each range. Calibration of your MODEL 1004 is best performed using calibration software available from Ascor.

When using multiple DMMs in a single chassis, the **SM40CAL.DAT** file must have a calibration record for each DMM. You can combine the unique calibration records of each DMM into one **SM40CAL.DAT** file using any ASCII text editor.

7.0 Warranty and Service

The DMM is warranted for a period of one year from date of purchase.

If your unit requires repair or calibration, contact your Ascor representative. There are no user serviceable parts within the DMM. Removal of any of the three external shields will invalidate your warranty. For in-warranty repairs, you must obtain a return authorization from Ascor prior to returning your unit.

8.0 Accessories

Several accessories are available for the MODEL 1004 DMMs, which can be purchased directly from Ascor, or one of its distributors or representatives. These include:

Basic DMM probes

DMM probe kit

Deluxe DMM probe set

`