DAQFlex

Message-based Firmware Specification

Document Revision 6
March 2012

MEASUREMENT COMPUTING™

Notice

Measurement Computing Corporation does not authorize any Measurement Computing Corporation product for use in life support systems and/or devices without prior written consent from Measurement Computing Corporation. Life support devices/systems are devices or systems which, a) are intended for surgical implantation into the body, or b) support or sustain life and whose failure to perform can be reasonably expected to result in injury. Measurement Computing Corporation products are not designed with the components required, and are not subject to the testing required to ensure a level of reliability suitable for the treatment and diagnosis of people.

Table of Contents

Message-based Firmware Specification

The message-based firmware architecture uses a String Message Based Interface (**SMBI**) to communicate with supported devices using a minimal number of device transactions.

For those interested in developing a driver to interface directly with the device, the source code for the DAQFlex software API may be used as a guide.

SMBI data types

Special formatting characters are returned with RAW data (except streaming data from bulk transfers) to indicate the following data types:

- char[64]: String data in a character array
- uint8: unsigned 8-bit value
- uint16: unsigned 16-bit value
- uint32: unsigned 32-bit value
- float32: float 32-bit value

Communication mechanism

The device enumerates as a USB device using an MCC driver.

Control transfers

- Control transfers to endpoint 0 are used to communicate with the device.
- The control transfer maximum buffer size is 64 bytes.
- The control transfers are vendor transfers.
- The command code is in the bRequest field.
  - SMBI uses the 0x80 data type for strings, and the 0x81 data type for RAW data, when supported.
- Parameters passed for OUT control transfers are sent during the data stage.
- The SMBI data stage returns RAW data, with LSB first.
- The DEV component message "DEV:DATATYPE=ENABLED" specifies whether raw data is enabled or disabled for returned data. When DEV:DATATYPE=ENABLED, the first byte indicates the data type returned. Refer to the RawData Notes for more information.
- The amount of data transferred must be specified in the `wLength` field.

- Bulk transfers from endpoint 6 IN are used for analog input scans. Data is in RAW format in 16-bits, with the 12-bit representation packed into the lower 16-bits.
- If a data overrun occurs, the device stalls the BULK IN endpoint (EP_6) when the AISCAN STALL property is set to ENABLE. To clear the stall, send AISCAN:RESET or start a new scan. The STALL property is disabled by default.
- Bulk transfers to endpoint 2 OUT are used for analog output scans. Data is transmitted in two bytes per channel, with the 12-bit representation packed into the lower 16-bits and transmitted in RAW format, resulting in an output of 0-4.096 V.
- The device is preloaded with up to 1024 bytes. The device will ACK while it is filling. When the four internal 256-byte buffers are filled, the device will NAK until an analog output scan is started.
- If a data underrun occurs, the device stalls the BULK OUT endpoint (EP_2) when the AOSCAN STALL property is set to ENABLE. To clear the stall, send AOSCAN:RESET or start a new scan. The STALL property is disabled by default.

Message-based control transfers

Digital I/O Commands

Command	Description	bRequest	Direction
StringMessage	Send string messages to the device	0x80	In/Out
RawData	Return raw data from the device	0x81	In

StringMessage — SMBI string message

This command reads/writes the string message components of a device. The device stores the various components and parameters to provide read back.

Output arguments

string char[64]; 64 character string including a NULL terminator.

Input arguments

None

Input response

string char[64]; 64 character response string including a NULL terminator.

Notes

- Invalid output strings result in a STALL to the USB Control Transfer, and the input response returns "INVALID".

RawData — Raw data with SMBI

This command reads the RAW data from the last StringMessage sent to the device. Data is only returned if the StringMessage has matching RAW data; otherwise a 0 length packet is returned.

Output arguments

N/A – output is not currently supported.

Input arguments

None

Input response

DATATYPE	0x03 – uint8 0x07 – uin16 0x09 – uint32 0x0A – float32 0xFF – Invalid
value	This depends upon the StringMessage sent, but will have the form of DATATYPE. Invalid StringMessage result in a 0 length packet, or if DATATYPE is set to ENABLE, one byte reading 0xFF.

Notes

- When "DEV:DATATYPE=ENABLE" the input response sets the first byte to the data type used. A valid SendMessage without a RAW return will return a 0 length packet. Only invalid StringMessages will return 0xFF.
- When "DEV:DATATYPE=DISABLE", the data will not include the DATATYPE byte.

Analog input data format

When performing analog input operations with the "AI" component, the return data is LSB (Least Significant Bit) justified with a value of 0 as Min Scale.

Example: The bit table below shows 12-bits of data acquired using a typical DAQFlex-supported device installed with firmware version 2.03 or later.

A/D Converter Values

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data	0	0	0	0	x	x	x	x	x	x	x	x	x	x	x	x

where *x* the actual conversion value.

Note

Some devices return data as 12-bit MSB (Most Significant Bit) justified, and require conversion to the LSB justified data format shown above. Refer to the device-specific information in the Hardware Reference chapter on page 21 for details.

Programming and developing applications

The firmware architecture uses a String Message Based Interface (SMBI) to communicate with DAQFlex series hardware. A minimal number of device transactions is required. Communication with the device is accomplished through the USB driver (mccusb, libusb, or other custom driver).

Device hardware installed on a Mac OS X system is configured with the libusb driver.

Refer to the Windows driver (mccusb) or the Linux driver (libusb) for an example of how to communicate with DAQFlex series hardware. You can also use the DAQFlex software API source code as a guide for developing a driver to interface directly with the hardware.

Notes

- Experience with CONTROL transfers for asynchronous communications, and BULK transfers for synchronous communications is recommended.
- Only experienced USB programmers should attempt to write a driver for use with the DAQFlex firmware.

See also

SMBI data types
Communication mechanism
Message-based control transfers

Updating device firmware

Note: To determine the firmware version currently installed on a device, run the DAQFlexFWLoader.exe provided with the DAQFlex for Windows software. The firmware version installed on a device displays in the **Device firmware version** field.

Perform the following procedure to update the firmware that is installed on a message-based DAQ device:

1. Go to the Measurement Computing Firmware Updates page at www.mccdaq.com/fwupdate.
2. Select the firmware version to download and click the **Download** button.
3. Right click on the firmware *.hex file and select the "Save As" option. Note that the option name may vary depending on the browser.
4. Save the *.hex file to the FirmwareImages subdirectory of the DAQFlex installation directory ("`C:\Program Files\Measurement Computing\DAQFlex For Windows\FirmwareImages`" by default).
5. Connect the message-based DAQ device to your computer.
6. Run the DAQFlex Firmware Loader utility from `C:\Program Files\Measurement Computing\DAQFlex For Windows`, and run **DAQFlexFWLoader.exe**.
7. From the **Device** drop down list select the device in which to update the firmware.
8. Click the **Load Firmware** button to start installing the firmware.

The progress bar updates as the firmware is installed on the device.

Once the firmware is installed, the **Status** field displays "Firmware update completed", and the **Device firmware version** field displays the current firmware version installed on the device. Click the **X** in the upper right corner of the dialog to exit the DAQFlex Firmware Loader utility.

Updating the FPGA configuration bitfile on a non-Windows system

For devices that support DEV:FPGACFG, a configuration bitfile must be loaded onto the device's FPGA before DAQ commands will work with the device.

The FPGA is configured automatically on a Windows operating system, and no action is required by the user. On Mac OS X and Linux systems, perform the procedure below to load or update the FPGA configuration bitfile on a device.

Get the status of the FPGA configuration

You can determine whether the device FPGA is configured by sending the "?DEV:FPGACFG" message. The device returns one of the following messages:

- "DEV:FPGA=CONFIGMODE" indicates that the FPGA is not configured,

  Perform the procedure below to load the FPGA configuration file onto the device.
- "DEV:FPGA=CONFIGURED" indicates that the device FPGA is currently configured.

  Perform the procedure below only when you want to update the configuration file.

Loading a FPGA configuration file onto the device (Linux and Mac OS X)

An FPGA configuration file has an `.rbf` extension and includes the name of the device. For example, the FPGA configuration file for the USB-1608G device is named *USB_1608g.rbf*.

FPGA configuration files are installed to the following location:

- **Mac OS X**: the FPGA configuration file is installed to `/usr/lib`.
- **Linux**: the FPGA configuration file is installed to `/usr/lib/daqflex`.

To load or update the FPGA configuration file on a device, complete the following steps:

1. Send "DEV:FPGACFG/<unlock_code>" to the device to put the device into FPGA Configuration Mode. The device response is "DEV:FPGACFG".
   Note: The <unlock_code> is device-specific. Refer to the DEV:FPGACFG message in the hardware-specific topics to determine the unlock code to set for a device. For example, to put the USB-1608G into FPGA configuration mode, send the message "DEV:FPGACFG/0xAD"
2. Stream the bitfile down to the device through endpoint 0:
3. Send the firmware command `FPGA_DATA` (0x51) to the device with 64 bytes from the FPGA bitfile. Specify the number of bytes to write with the `wLength` field.

   Repeat until the entire FPGA bitfile has been sent. The FPGA bitfile is streamed down to the device through Endpoint 0.

The device automatically exits Configuration Mode after the FPGA image file is successfully downloaded.

Example code

The following code shows how to load the FPGA configuration bitfile onto a USB-1608G device. The code below is written in C++/pseudocode, but it can be written in any language.

```
file = open(USB_1608g.rbf);
size = getSize(file);
while (size > 0);
{
    count = file read(buffer, 64);
    device = ControlTransfer(RT_out, 0x51, count, buffer);
    size = size – count;
}
file close();
```

Calibrating a device

The message-based firmware provides messages that let you calibrate the analog inputs or outputs on a device.

Calibrating analog inputs

To modify the analog input calibration settings, complete the following steps:

1. Send the "AICAL:UNLOCK" message to unlock the AICAL component and put the device into Calibrate Mode.
   When the AICAL component is unlocked, sending a message to any component other than AICAL will lock the AICAL component.
2. Send the "AICAL:REF=*value*" message to set the calibration voltage.
   Refer to the device-specific information to determine the calibration voltages supported by the device.
   The calibration voltage value is stored in EEPROM and can be retrieved with the "?AICAL:REFVAL" query.
3. Send the "AICAL{ch}:RANGE=*value*" message to set the channel and range value to calibrate.
4. Send the "AICAL{ch}:SLOPE=*value*" message to set the slope for the current range setting.
5. Send the "AICAL{ch}:OFFSET=*value*" message to set the offset for the current range setting.
6. Send the message "AICAL:LOCK" to lock the AICAl component and exit Calibration Mode.
   The device returns INVALID if any AICAL property message is sent while the AICAL component is locked.

Note: The channel {ch} message variable is ignored on devices with channels that are not individually configurable.

Refer to the Analog input calibration messages on page 12 for details about the AI calibration messages .

Calibrating analog outputs

To modify the analog output calibration settings, complete the following steps:

1. Send the "AOCAL:UNLOCK" message to lock the AOCAL component and put the device into Calibrate Mode.
   When the AOCAL component is unlocked, sending a message to any component other than AOCAL will lock the AOCAL component.
2. Send the "AOCAL{ch}:SLOPE=*value*" message to set the slope for the current range setting.
3. Send the "AOCAL{ch}:OFFSET=*value*" message to set the offset for the current range setting.
4. Send the "AOCAL{ch}:VALUE=*value*" message to set the value of the AO channel being calibrated.
5. Send the message "AOCAL:LOCK" to lock the AOCAl component and exit Calibration Mode.
   The device returns INVALID if any AOCAL property message is sent while the AOCAL component is locked.

Note: The channel {ch} message variable is ignored on devices with channels that are not individually configurable.

Refer to the Analog output calibration messages on page 16 for details about the AO calibration messages.

Analog input calibration messages

The AI and AICAL components provide messages for calibrating the analog inputs on a device. Refer to the device-specific information in the Hardware Reference chapter for the component properties and commands supported by each DAQ device.

AI component calibration properties

ADCAL/START, ADCAL/STATUS

ADCAL/START

- Start the A/D internal calibration.

  Message "AI:ADCAL/START"

  Response "AI:ADCAL/START"

ADCAL/STATUS

- Get the status of the A/D internal calibration.

  Message "?AI:ADCAL/STATUS"

  Response "AI:ADCAL/STATUS=*value*"

  value RUNNING, IDLE

AICAL component calibration properties

LOCK, MODE, OFFSET, RANGE, REF, REFVAL, RES, SLOPE, TCCAL, UNLOCK, VALUE

LOCK

- Lock the AICAL component.

  Message "AICAL:LOCK"

  Response "AICAL:LOCK"

  Note The device returns INVALID if any AICAL property message is accessed while the AICAL component is locked.

MODE

- Set the calibration channel mode.

  Message "AICAL:MODE=*value*"

  Response "AICAL:MODE"

  value AIOFFSET, AIGAIN, TCOFFSET, TCGAIN/POS, TCGAIN/NEG

  Example "AICAL:MODE=TCOFFSET"

- Get the calibration channel mode.

  Message "?AICAL:MODE"

  Response "AICAL:MODE=*value*"

  value AIOFFSET, AIGAIN, TCOFFSET, TCGAIN/POS, TCGAIN/NEG

OFFSET

- Set the offset in hexadecimal bytes for the current AICAL:RANGE setting.

  Message "AICAL{ch}:OFFSET/HEX=*hex value*"
  or
  "AICAL{ch}:OFFSET=0x*hex value*"

  Refer to the device-specific information in the Hardware Reference chapter to see whether "/HEX" is supported.

  Response "AICAL{ch}:OFFSET"

  ch The channel number.

  hex value The calibration offset in hexadecimal bytes.

  Example "AICAL{0}:OFFSET/HEX=0x84B575AF"

  Note The hex value is used to preserve the precision of the double precision floating point value.

- Get the offset in hexadecimal bytes for the current AICAL:RANGE setting.

  Message "?AICAL{ch}:OFFSET/HEX"
  or
  "?AICAL{ch}:OFFSET"

  Refer to the device-specific information in the Hardware Reference chapter to see whether "/HEX" is supported.

  Response "AICAL{ch}:OFFSET=0x*hex value*"

  ch The channel number.

  hex value The calibration offset in hexadecimal bytes.

RANGE

- Set the range to calibrate.

  Message "AICAL:RANGE=*value*"

  Response "AICAL:RANGE"

  value The range value.

  Example "AICAL:RANGE=BIP10V"

- Get the range to calibrate.

  Message "?AICAL:RANGE"

  Response "AICAL:RANGE=*value*"

  value The range value.

REF

- Set the calibration reference voltage.

  Message "AICAL:REF=*value*"

  Response "AICAL:REF"

  value The voltage value.

  Example "AICAL:REF=+1.250034V"

  Note The voltage value is stored in EEPROM.

REFVAL

- Get the actual measured value of the currently set calibration reference voltage.

  Message "?AICAL:REFVAL"

  Response "AICAL:REFVAL=*value*"

  value The measured voltage value.

  Note The voltage value is retrieved from the value stored in EEPROM.

- Get the actual measured hex value of the currently set calibration reference voltage.

  Message "?AICAL:REFVAL/HEX"

  Response "AICAL:REFVAL=0x*hex value*"

  hex value The measured voltage hex value.

  Note Omitting "/HEX" from the query results in a loss of precision in the value returned.

RES

- Get the A/D resolution.

  Message "?AICAL:RES"

  Response "AICAL:RES=*value*"

  value The A/D resolution.

SLOPE

- Set the slope for the current AICAL:RANGE setting.

  Message "AICAL{ch}:SLOPE=*value*"

  Response "AICAL{ch}:SLOPE"

  ch The channel number.

  value The calibration slope.

  Example "AICAL{0}:SLOPE=–0.0025"

- Set the slope in hexadecimal bytes for the current AICAL:RANGE setting.

  Message "AICAL{ch}:SLOPE/HEX=*hex value*"

  Response "AICAL{ch}:SLOPE"

  ch The channel number.

  hex value The calibration slope in hexadecimal bytes.

  Example "AICAL{0}:SLOPE/HEX=0x14B525A3"

  Note The hex value is used to preserve the precision of the double precision floating point value.

- Get the slope in hexadecimal bytes for the current AICAL:RANGE setting.

  Message "?AICAL{ch}:SLOPE/HEX"
  or
  "?AICAL{ch}:SLOPE"

  Refer to the device-specific topics to see whether "/HEX" is supported.

  Response "AICAL{ch}:SLOPE=0x*hex value*"

  ch The channel number.

  hex value The calibration slope in hexadecimal bytes.

TCCAL

- Enables or disables the TC Cal Measurement option.

  Message "AICAL:TCCAL=*value*"

  Response "AICAL:TCCAL"

  value ENABLE, DISABLE

  Example "AICAL:TCCAL=ENABLE"

- Get the value for the TC Cal Measurement option.

  Message "?AICAL:TCCAL"

  Response "AICAL:TCCAL=*value*"

  value ENABLE, DISABLE

UNLOCK

- Unlock the AICAL component.

  Message "AICAL:UNLOCK"

  Response "AICAL:UNLOCK"

  Note The AICAL component must be unlocked in order to change calibration settings on a device's analog inputs.

  When AICAL is unlocked, sending a message to any component other than AICAL will lock the AICAL component.

VALUE

- Get the value of the analog input channel being calibrated.

  Message "?AICAL{ch}:VALUE"

  Response "AICAL{ch}:VALUE=*value*"

  ch The AI channel being calibrated.

  value The AI channel value in counts.

  Example "?AICAL{0}:VALUE"

Analog output calibration messages

The AOCAL component provides messages for calibrating the analog outputs on a device. Refer to the device-specific information in the Hardware Reference chapter for the component properties and commands supported by each DAQ device.

AOCAL component calibration properties

AIOFFSET, AIRANGE, AIRES, AISLOPE, AIVALUE, LOCK, OFFSET, RES, SLOPE, UNLOCK, VALUE

AIOFFSET

- Get the offset for the current AICAL:RANGE setting.

  Message "?AOCAL{ch}:AIOFFSET/HEX"

  Response "AOCAL{ch}:AIOFFSET=0x*hex value*"

  ch The analog output channel.

  hex value The offset currently set for AICAL:RANGE.

  Example "AOCAL{0}:AIOFFSET=0x84B575AF"

  Note The hex value is used to preserve the precision of the double precision floating point value.

AIRANGE

- Get the analog input range.

  Message "?AOCAL{ch}:AIRANGE"

  Response "AOCAL{ch}:AIRANGE=*value*"

  ch The analog output channel.

  value BIP10V.

  Example "AOCAL{0}:AIRANGE=BIP10V"

AIRES

- Get the resolution of the A/D used for calibrating the D/A.

  Message "?AOCAL:AIRES"

  Response "AOCAL:AIRES=*value*"

  value The A/D resolution.

  Example "AOCAL:AIRES=S24"

  Note The first character in the value returned indicates that the value is signed.

AISLOPE

- Get the slope for the current AICAL:RANGE setting.

  Message "?AOCAL{ch}:AISLOPE/HEX"

  Response "AOCAL{ch}:AISLOPE=0x*hex value*"

  ch The analog output channel.

  hex value The slope currently set for AICAL:RANGE.

  Example "AOCAL{0}:AISLOPE=0x14B525A3"

  Note The hex value is used to preserve the precision of the double precision floating point value.

AIVALUE

- Get the value of the analog input channel being calibrated.

  Message "?AOCAL{ch}:AIVALUE"

  Response "AOCAL{ch}:AIVALUE=*value*"

  ch The AI channel being calibrated.

  value The AI channel value in counts.

  Note When this message is sent, the device switches the internal CalConfig to the DAC channel that is specified in the message.

LOCK

- Lock the AOCAL component.

  Message "AOCAL:LOCK"

  Response "AOCAL:LOCK"

  Note The device returns INVALID if any AOCAL property message is accessed while the AOCAL component is locked.

OFFSET

- Set the offset in hexadecimal bytes for the current AICAL:RANGE setting.

  Message "AICAL{ch}:OFFSET/HEX=0x*hex value*"
  or
  "AICAL{ch}:OFFSET=0x*hex value*"

  Refer to the device-specific topics to see whether "/HEX" is supported.

  Response "AICAL{ch}:OFFSET"

  ch The channel number.

  hex value The calibration offset in hexadecimal bytes.

  Example "AICAL{0}:OFFSET/HEX=0x84B575AF"

  Note The hex value is used to preserve the precision of the double precision floating point value.

- Get the offset in hexadecimal bytes for the current AICAL:RANGE setting.

  Message "?AICAL{ch}:OFFSET/HEX"
  or
  "?AICAL{ch}:OFFSET"

  Refer to the device-specific topics to see whether "/HEX" is supported.

  Response "AICAL{ch}:OFFSET=0x*hex value*"

  ch The channel number.

  hex value The calibration offset in hexadecimal bytes.

RES

- Get the D/A resolution.

  Message "?AICAL:RES"

  Response "AICAL:RES=*value*"

  value The A/D resolution.

SLOPE

- Set the slope for the current AICAL:RANGE setting.

  Message "AICAL{ch}:SLOPE=*value*"

  Response "AICAL{ch}:SLOPE"

  ch The channel number.

  value The calibration slope.

  Example "AICAL{0}:SLOPE=–0.0025"

- Set the slope in hexadecimal bytes for the current AICAL:RANGE setting.

  Message "AICAL{ch}:SLOPE/HEX=0x*hex value*"

  Response "AICAL{ch}:SLOPE"

  ch The channel number.

  hex value The calibration slope in hexadecimal bytes.

  Example "AICAL{0}:SLOPE/HEX=0x14B525A3"

  Note The hex value is used to preserve the precision of the double precision floating point value.

- Get the slope in hexadecimal bytes for the current AICAL:RANGE setting.

  Message "?AICAL{ch}:SLOPE/HEX"
  or
  "?AICAL{ch}:SLOPE"

  Refer to the device-specific topics to see whether "/HEX" is supported.

  Response "AICAL{ch}:SLOPE=0x*hex value*"

  ch The channel number.

  hex value The calibration slope in hexadecimal bytes.

UNLOCK

- Unlock the AOCAL component.

  Message "AOCAL:UNLOCK"

  Response "AOCAL:UNLOCK"

  Note The AOCAL component must be unlocked in order to change calibration settings on a device's analog outputs.

  When AOCAL is unlocked, sending a message to any component other than AOCAL will lock the AOCAL component.

VALUE

- Set the value of the analog output channel being calibrated.

  Message "?AOCAL{ch}:VALUE=*value*"

  Response "AOCAL{ch}:VALUE"

  ch The AO channel being calibrated.

  value The AO channel value in counts.

  Example "AOCAL{0}:VALUE=4095"

- Get the value of the analog output channel being calibrated.

  Message "?AOCAL{ch}:VALUE"

  Response "AOCAL{ch}:VALUE=*value*"

  ch The AO channel being calibrated.

  value The AO channel value in counts.

Hardware Reference

Select your DAQFlex-supported device below for the components and programming messages supported by the firmware.

- USB-1608FS-Plus
- USB-1608G Series
- USB-2408 Series
- USB-2001-TC
- USB-7202
- USB-7204

USB-1608FS-Plus

Use the components below to set or get device properties.

Component	Supported Property/Command	Set/Get	Supported Values
AI	RES	Get	U16
AI{ch}	OFFSET	Get	4-byte floating point numeric
	RANGE	Set/Get	BIP10V, BIP5V, BIP2V, BIP1V
	SLOPE	Get	4-byte floating point numeric
	VALUE	Get	0 to 7; Unsigned integer numeric
AIQUEUE	CLEAR	Set	
	COUNT	Get	0 to 7 elements
AIQUEUE{element}	CHAN	Set/Get	element: 0 to 63 value: 0 to 15
	RANGE	Set/Get	element: 0 to 7 value: BIP10V, BIP5V, BIP2V, BIP1V
AISCAN	EXTPACER	Set/Get	ENABLE/MASTER, ENABLE/SLAVE, DISABLE
	HIGHCHAN	Set/Get	0 to 7 single-ended
	LOWCHAN	Set/Get	0 to 7 single-ended
	QUEUE	Set/Get	ENABLE, DISABLE
	RATE	Set/Get	0.009 S/s to 100 kS/s
	RANGE	Set/Get	BIP10V, BIP5V, BIP2V, BIP1V
	Range{ch}		
	RESET	Set	
	SAMPLES	Set/Get	0 to N (0 = continuous scan; N = 32-bit FIFO)
	STALL	Set/Get	ENABLE, DISABLE
	START	Set	
	STATUS	Get	IDLE, RUNNING, OVERRUN
	STOP	Set	
	TRIG	Set/Get	ENABLE, DISABLE

Component	Supported Property/Command	Set/Get	Supported Values
	XFRMODE	Set/Get	BLOCKIO, SINGLEIO, BURSTIO
AITRIG	TYPE	Set/Get	EDGE/RISING, EDGE/FALLING
CTR{ch}	VALUE	Get	0 to 4,294,967,295
		Set	0
DEV	DATATYPE	Set/Get	ENABLE, DISABLE
	FLASHLED	Set	0 to 255
	FWV	Get	Firmware version of the device MM.mm (M = major; m = minor)
	ID	Set/Get	Up to 57 characters
	MFGCAL	Get	yyyy-mm-dd HH:MM:SS Year as yyyy; 20xx Month as mm; 01 to 12 Day as dd; 01 to 31 Hour as HH; 01 to 23 Minute as MM; 01 to 59 Second as SS; 01 to 59
	MFGSER	Get	Up to 8 numeric characters
	RESET	Set	DEFAULT
DIO{port}	DIR	Set/Get	IN, OUT
	LATCH	Set/Get	0 to 255
	VALUE	Set/Get	0 to 255
DIO{port/bit}	DIR	Set/Get	IN, OUT
	LATCH	Set/Get	port number: 0 bit number: 0 to 7 port value: 0 to 255 bit value: 0, 1
	VALUE	Set/Get	port number: 0 bit number: 0 to 7 port value: 0 to 255 bit value: 0, 1

Hardware features

- 8 single-ended, 16-bit, simultaneous analog input channels, numbered 0 to 7.
  Analog voltage input ranges:
  - ±10V
  - ±5V
  - ±2V
  - ±1V
- Sampling rate:
  - 100 kS/s max rate for one channel
  - 400 kS/s max aggregate rate, streaming. Limited by USB transfer speed; 800 kS/s max to 32K sample FIFO
  - Channel-gain queue:
  - Eight elements - one gain element per channel.
- 8 DIO, individually configurable

- External digital trigger
- Bidirectional external clock
- 32-bit event counter
- 1024 bytes of nonvolatile EEPROM memory; used for storing configuration information, calibration data, and user data.

USB-1608G Series

The USB-1608G Series includes the following devices:

- USB-1608G
- USB-1608GX
- USB-1608GX-2AO

Use the components below to set or get device properties.

Component	Supported Property/Command	Set/Get	Supported Values
AI	RES	Get	A/D resolution S24 (24-bit signed value)
AI{ch}	CHMODE	Get	SE, DIFF, MIXED
	OFFSET	Get	4-byte floating point numeric
	RANGE	Set/Get	BIP10V, BIP5V, BIP2V, BIP1V
	SLOPE	Get	4-byte floating point numeric
	VALUE	Get	Counts
AICAL	LOCK	Set	
	RANGE	Set/Get	BIP10V, BIP5V, BIP2V, BIP1V
	REF	Set/Get	0.0V, +1.0V, –1.0V, +2.0V, –2.0V, +5.0V, –5.0V, +10.0V, –10.0V
	REFVAL	Get	The measured calibration reference voltage stored in EEPROM memory.
	RES	Get	A/D resolution S24 (24-bit signed value)
	UNLOCK	Set	
AICAL{ch}	OFFSET	Set/Get	4-byte floating point numeric. The channel is ignored.
	SLOPE	Set/Get	4-byte floating point numeric. The channel is ignored.
	VALUE	Get	The value in counts of the AI channel being calibrated.
AIQUEUE	CLEAR		
	COUNT	Get	0 to 16 elements
AIQUEUE{element}	CHAN	Set/Get	0 to 15 single-ended, 0 to 7 differential
	CHANMODE	Set/Get	SE, DIFF
	RANGE	Set/Get	BIP10V, BIP5V, BIP2V, BIP1V

Component	Supported Property/Command	Set/Get	Supported Values
AISCAN	BURSTMODE	Set/Get	ENABLE, DISABLE
	DEBUG	Set	ENABLE, DISABLE
	EXTPACER	Set/Get	ENABLE, DISABLE
	HIGHCHAN	Set/Get	0 to 15 single-ended, 0 to 7 differential
	LOWCHAN	Set/Get	0 to 15 single-ended, 0 to 7 differential
	QUEUE	Set/Get	ENABLE, DISABLE, RESET
	RANGE	Set/Get	BIP10V, BIP5V, BIP2V, BIP1V
	RANGE{ch}		
	RATE	Set/Get	USB-1608G: 0.01 Hz to 250,000 Hz (1 channel) USB-1608GX: 0.01 Hz to 500,000 Hz (1 channel) USB-1608GX-2AO: 0.01 Hz to 500,000 Hz (1 channel)
	RESET	Set	
	SAMPLES	Set/Get	0 to N (0 = continuous scan; N = 32-bit)
	STALL	Set/Get	ENABLE, DISABLE
	START		
	STATUS	Get	IDLE, RUNNING, OVERRUN
	STOP		
	TRIG	Set/Get	ENABLE, DISABLE
	XFRMODE	Set/Get	BLOCKIO, SINGLEIO
AITRIG	REARM	Set/Get	ENABLE, DISABLE
	TYPE	Set/Get	EDGE/{condition}, LEVEL/{condition} condition: RISING, FALLING when TYPE is EDGE HIGH, LOW when TYPE is LEVEL
AO[1]	RES	Get	D/A resolution U16 (unsigned 16-bitinteger)
AO{ch}[1]	OFFSET	Get	4-byte floating point numeric
	RANGE	Get	BIP10V
	SLOPE	Get	4-byte floating point numeric
	VALUE	Set/Get	0 to 65,535 counts
AOCAL	AIRES	Get	S24 (24-bit signed value)
	LOCK	Set	
	RES	Get	U16 (16-bit unsigned value)
	UNLOCK	Set	

Component	Supported Property/Command	Set/Get	Supported Values
AOCAL{ch}	AIOFFSET	Get	Double precision floating point numeric (hex bytes). The channel is ignored.
	AIRANGE	Get	BIP10V
	AISLOPE	Get	Double precision floating point numeric (hex bytes). The channel is ignored.
	AIVALUE	Get	Counts
	OFFSET	Set/Get	Double precision floating point numeric
	SLOPE	Set/Get	Double precision floating point numeric
	VALUE	Set/Get	0 to 65,535 counts
AOSCAN[1]	EXTPACER	Set/Get	ENABLE, DISABLE
	HIGHCHAN	Set/Get	0 to 1
	LOWCHAN	Set/Get	0 to 1
	RANGE	Get	BIP10V
	RATE	Set/Get	1 kHz to 500 kHz (1 channel)
	SAMPLES	Set/Get	0 to N (0 = continuous scan; N = 32-bit)
	STALL	Set/Get	ENABLE, DISABLE
	START	Set	
	STATUS	Get	IDLE, RUNNING, UNDERRUN
	STOP	Set	
CTR{ch}	START		
	STOP		
	VALUE	Set	0
		Get	0 – 4,294,967,295
DEV	DATATYPE	Set/Get	ENABLE, DISABLE
	FLASHLED	Set	0 - 255
	FPGAV	Get	FPGA firmware version
	FPGACFG	Set	0xAD
		Get	CONFIGMODE, CONFIGURED
	FWV	Get	Firmware version; MM.mm (M = major; m = minor)
	ID	Set/Get	Up to 57 characters
	MFGCAL	Get	yyyy-mm-dd HH:MM:SS
	MFGSER	Get	Up to 8 numeric characters
	RESET		DEFAULT, SYSTEM
	TEMP	Get	ºC
DIO{port}		Get	8
	DIR	Set/Get	IN, OUT
	LATCH	Set/Get	0 to 255
	VALUE	Set/Get	0 to 255

Component	Supported Property/Command	Set/Get	Supported Values
DIO{port/bit}	DIR	Set/Get	IN, OUT
	LATCH	Set/Get	port number: 0 bit number: 0 to 7 port value: 0 to 255 bit value: 0, 1
	VALUE	Set/Get	port number: 0 bit number: 0 to 7 port value: 0 to 255 bit value: 0, 1
TMR{ch}	DELAY	Set/Get	0.00003125 mS to 67110 mS
	DUTYCYCLE	Set/Get	0 to 100%
	IDLESTATE	Set/Get	LOW, HIGH
	PERIOD	Set/Get	0.00003125 mS to 67110 mS
	PULSECOUNT	Set/Get	0 to 4294967295
	START	Set	
	STOP	Set	
[1] Analog output is supported on the USB-1608GX-2AO only.			

Hardware features

- 16 analog input channels, numbered 0 to 15.
- Analog input mode is configurable for single-ended (16 channels) or differential (8 channels).
  - Analog input ranges:
  - ±10V
  - ±5V
  - ±2V
  - ±1V
- 2 analog output channels, numbered 0 to 1 (USB-1608GX-2AO only)
- Analog output range is fixed at ±10V.
- 1 digital port (8 bits)
- Each bit is individually configurable as input or output.
- 1 timer output channel
- 1 external trigger input
- External pacer input/output
- This feature allows multiple devices to acquire synchronized samples.
- 1024 bytes of nonvolatile EEPROM memory; used for storing configuration information, calibration data, and user data.
- *RATE* takes a float value
- An error is generated if the input scan rate requested is less than the device's minimum sampling rate or greater than the device's maximum sampling rate.

- Calibration operations
  - AICAL:LOCK locks the AICAL component from performing calibration operations on the analog inputs. In order to calibrate the analog inputs, you must first unlock the component using the AICAL:UNLOCK message.

  - AOCAL:LOCK locks the AOCAL component from performing calibration operations on the analog outputs. In order to calibrate the analog outputs, you must first unlock the component using the AOCAL:UNLOCK message.

- Refer to Analog input calibration messages and Analog output calibration messages for the calibration messages provided by the firmware.

USB-2001-TC

Use the components below to set or get device properties.

Component	Supported Property/Command	Set/Get	Supported Values
AI	CJC/format	Get	CJC/DEGC, CJC/DEGF, CJC/KELVIN
	OFFSET	Get	Floating point numeric
	SENSOR	Set/Get	TC/char
	SLOPE	Get	Floating point numeric
	STATUS	Get	BUSY, ERROR, READY
	VALUE	Get	Unsigned integer numeric
AI{ch}	RANGE	Set/Get	BIP73.125E-3V (±0.073125 volts) BIP146.25E-3V (±0.14625 volts)
DEV	FLASHLED	Set	0 to 255
	FWV	Get	Firmware version
	ID	Set/Get	Up to 57 characters
	MFGCAL	Get	yyyy-mm-dd HH:MM:SS
	MFGCAL{YEAR}	Get	Year as yyyy; 20xx
	MFGCAL{MONTH}	Get	Month as mm; 01 to 12
	MFGCAL{DAY}	Get	Day as dd; 01 to 31
	MFGCAL{HOUR}	Get	Hour as HH; 0 to 23
	MFGCAL{MINUTE}	Get	Minute as MM; 1 to 59
	MFGCAL{SECOND}	Get	Second as SS; 1 to 59
	MFGSER	Get	Up to 8 numeric characters
	RESET		DEFAULT, SYSTEM

Hardware features

- One analog input channel, numbered 0
- Supports thermocouple types B, E, J, K, N, R, S, and T
- Possible gain ranges:
  - ±146.25 mV (not calibrated)
  - ±73.125 mV
- 1024 bytes of nonvolatile FLASH program memory; used for storing configuration information, calibration data, and user data.

USB-2408 Series

The USB-2408 Series includes the following devices:

- USB-2408
- USB-2408-2AO

Use the components below to set or get device properties.

Component	Supported Property/Command	Set/Get	Supported Values/Notes
AI	ADCAL/START	Set	
	ADCAL/STATUS	Get	RUNNING, IDLE
	RES	Get	A/D resolution S24 (24-bit signed integer)
AI{ch}	CHMODE	Set/Get	SE, DIFF, TC/OTD, TC/NOOTD SE is always returned for channels 8 to 15.
	CJC	Get	DEGC, DEGF, KELVIN
	DATARATE	Set/Get	3750, 2000, 1000, 500, 100, 60, 50, 25, 10, 5, 2.5 (S/s)
	OFFSET	Get	4-byte floating point numeric
	RANGE	Set/Get	BIP10V, BIP5V, BIP2.5V, BIP1.25V, BIP0.625V, BIP0.312V, BIP0.156V, BIP78.125E-3V
	SENSOR	Set/Get	TC/B, TC/E, TC/J, TC/K, TC/N, TC/R, TC/S, TC/T
	SLOPE	Get	4-byte floating point numeric
	VALUE	Get	Unsigned integer numeric
AICAL	LOCK	Set	
	MODE	Set/Get	ENABLE, DISABLE
	RANGE	Set/Get	BIP10V, BIP5V, BIP2.5V, BIP1.25V, BIP0.625V, BIP0.312V, BIP0.156V, BIP78.125E-3V
	REF	Set/Get	+78.125E-3V, –78.125E-3V, +156.25E-3V, –156.25E-3V, +312.5E-3V, –312.5E-3, +625.0E-3V, -625.0E-3V, +1.25V, –1.25V, +2.5V, –2.5V, +5.0V, –5.0V, +10.0V, –10.0V
	REFVAL/HEX	Get	The measured calibration reference voltage (hex bytes) stored in EEPROM memory.
	RES	Get	A/D resolution S24 (24-bit signed value)
	UNLOCK	Set	
AICAL{ch}	OFFSET/HEX	Set/Get	Double precision floating point numeric (hex bytes); The channel is ignored.
	SLOPE/HEX	Set/Get	Double precision floating point numeric (hex bytes). The channel is ignored.
	TCCAL	Set/Get	ENABLE, DISABLE
	VALUE	Get	The value in counts of the AI channel being calibrated.

Component	Supported Property/Command	Set/Get	Supported Values/Notes
AIQUEUE	CLEAR	Set	
	COUNT	Get	0 to 64 elements
AIQUEUE{element}	CHAN	Set/Get	element: 0 to 63 value: 0 to 15
	DATARATE	Set/Get	3750, 2000, 1000, 500, 100, 60, 50, 25, 10, 5, 2.5 (S/s)
	CHANMODE	Set/Get	element: 0 to 63 value: SE, DIFF, TC/OTD, TC/NOOTD
	RANGE	Set/Get	element: 0 to 63 value: BIP10V, BIP5V, BIP2.5V, BIP1.25V, BIP0.625V, BIP0.312V, BIP0.156V, BIP78.125E-3V
AISCAN	HIGHCHAN	Set/Get	0 to 15 single-ended, 0 to 7 differential
	LOWCHAN	Set/Get	0-15 single-ended, 0 to 7 differential
	QUEUE	Set/Get	ENABLE, DISABLE
	RATE	Set/Get	2.5 Hz to 1102.94 Hz (1 channel)
	RANGE, RANGE{ch}	Set/Get	BIP10V, BIP5V, BIP2.5V, BIP1.25V, BIP0.625V, BIP0.312V, BIP0.156V, BIP78.125E-3V
	RESET	Set	
	SAMPLES	Set/Get	0 to N (0 = continuous scan; N = 32-bit)
	STALL	Set/Get	ENABLE, DISABLE
	START	Set	
	STATUS	Get	IDLE, RUNNING, OVERRUN
	STOP	Set	
	XFRMODE	Set/Get	BLOCKIO, SINGLEIO
AO[1]	RES	Get	D/A resolution U16 (unsigned 16-bitinteger)
	UPDATE		
AO{ch}[1]	OFFSET	Get	4-byte floating point numeric
	RANGE	Get	BIP10V
	REG	Set/Get	0 to 4095
	SLOPE	Get	4-byte floating point numeric
	VALUE	Set	0 to 4095
AOCAL	AIRES	Get	S24 (24-bit signed value)
	LOCK	Set	
	RES	Get	U16 (16-bit unsigned value)
	UNLOCK	Set	
AOCAL{ch}	AIOFFSET	Get	Double precision floating point numeric (hex bytes). The channel is ignored.
	AIRANGE	Get	BIP10V

Component	Supported Property/Command	Set/Get	Supported Values/Notes
	AISLOPE	Get	Double precision floating point numeric (hex bytes). The channel is ignored.
	AIVALUE	Get	Counts
	OFFSET	Set/Get	Double precision floating point numeric
	SLOPE	Set/Get	Double precision floating point numeric
	VALUE	Set/Get	0 to 65,535 counts
AOSCAN[1]	HIGHCHAN	Set/Get	0 to 1
	LOWCHAN	Set/Get	0 to 1
	RANGE, RANGE{ch}	Get	BIP10V
	RESET	Set	
	RATE	Set/Get	1 Hz to 1000 Hz (1 channel)
	SAMPLES	Set/Get	0 to N (0 = continuous scan; N = 32-bit)
	STALL	Set/Get	ENABLE, DISABLE
	START	Set	
	STATUS	Get	IDLE, RUNNING, UNDERRUN
	STOP	Set	
CTR{ch}	START	Set	
	STOP	Set	
	VALUE	Get	0 to 4,294,967,295
		Set	0
DEV	DATATYPE	Set/Get	ENABLE, DISABLE
	FLASHLED	Set	0 to 255
	FWV	Get	MM.mm (M = major; m = minor)
	ID	Set/Get	Up to 57 characters
	MFGCAL	Get	yyyy-mm-dd HH:MM:SS Year as yyyy; 20xx Month as mm; 01 to 12 Day as dd; 01 to 31 Hour as HH; 01 to 23 Minute as MM; 01 to 59 Second as SS; 01 to 59
	MFGSER	Get	Up to 8 numeric characters
	RESET	Set	DEFAULT
	STATUS/ISO	Get	Isolated microcontroller status READY, NOTREADY
DIO{port}		Get	8
	DIR	Set/Get	IN, OUT
	LATCH	Set/Get	0 to 255
	VALUE	Set/Get	0 to 255

Component	Supported Property/Command	Set/Get	Supported Values/Notes
DIO{port/bit}	DIR	Set/Get	IN, OUT
	LATCH	Set/Get	port number: 0 bit number: 0 to 7 port value: 0 to 255 bit value: 0, 1
	VALUE	Set/Get	port number: 0 bit number: 0 to 7 port value: 0 to 255 bit value: 0, 1
[1] Analog output is supported on the USB-2408-2AO only.			

Hardware features

- 16 analog input channels, numbered 0 to 15.
  - Analog input mode is configurable for single-ended (16 channels) or differential (8 channels).
  - Thermocouple mode requires a differential configuration.
  - Analog voltage input ranges:
  - ±10V
  - ±5V
  - ±2.5V
  - ±1.25V
  - ±0.625V
  - ±0.3125V
  - ±0.15625V
  - ±0.078125V
  - Analog thermocouple input range is fixed at ±0.078125V.
- 2 analog output channels, numbered 0 to 1 (USB-2408-2AO only).
- Analog output range is fixed at ±10V
- 1 digital input/output port (8 bits).
- 1024 bytes of nonvolatile EEPROM memory; used for storing configuration information, calibration data, and user data.
- *RATE* takes a float value. An error is generated if *value* set is less than the device's minimum sampling rate or greater than the device's maximum sampling rate.
- Calibration operations
  - AICAL:LOCK locks the AICAL component from performing calibration operations on the analog inputs. In order to calibrate the analog inputs, you must first unlock the component using the AICAL:UNLOCK message.
  - AOCAL:LOCK locks the AOCAL component from performing calibration operations on the analog outputs. In order to calibrate the analog outputs, you must first unlock the component using the AOCAL:UNLOCK message.
- Refer to Analog input calibration messages and Analog output calibration messages for the calibration messages provided by the firmware.

USB-7202

Use the components below to set or get device properties.

Component	Supported Property/Command	Set/Get	Supported Values
AI{ch}	OFFSET	Get	4-byte floating point numeric
	RANGE	Set/Get	BIP10V, BIP5V, BIP2V, BIP1V
	SLOPE	Get	4-byte floating point numeric
	VALUE	Get	Counts
AISCAN	DEBUG	Set	ENABLE, DISABLE
	EXTPACER	Set/Get	ENABLE/MASTER, ENABLE/SLAVE, DISABLE
	HIGHCHAN	Set/Get	0 to 7
	LOWCHAN	Set/Get	0 to 7
	RANGE	Set/Get	BIP10V, BIP5V, BIP2V, BIP1V
	RANGE{ch}		
	SAMPLES	Set/Get	0 to N (0 = continuous scan; N = 32-bit)
	RATE	Set/Get	0.596 to 50,000 Hz (1 channel)
	STALL	Set/Get	ENABLE, DISABLE
	START		
	STATUS	Get	IDLE, RUNNING, OVERRUN, INTERRUPTED
	STOP		
	TRIG	Set/Get	ENABLE, DISABLE
	XFRMODE	Set/Get	BLOCKIO, SINGLEIO, BURSTIO
AITRIG	TYPE	Set/Get	EDGE/RISING, EDGE/FALLING
CTR	START		
	STOP		
	VALUE	Set	0
		Get	0 to 4,294,967,295
DEV	DATATYPE	Set/Get	ENABLE, DISABLE
	FLASHLED	Set	0 to 255
	FWV	Get	MM.mm (M = major; m = minor)
	ID	Set/Get	Up to 57 characters
	MFGCAL	Get	yyyy-mm-dd HH:MM:SS
	MFGCAL{YEAR}	Get	Year as yyyy; 20xx
	MFGCAL{MONTH}	Get	Month as mm; 01 to 12
	MFGCAL{DAY}	Get	Day as dd; 01 to 31
	MFGCAL{HOUR}	Get	Hour as HH; 0 to 23
	MFGCAL{MINUTE}	Get	Minute as MM; 1 to 59
	MFGCAL{SECOND}	Get	Second as SS; 1 to 59
	MFGSER	Get	Up to 8 numeric characters

Component	Supported Property/Command	Set/Get	Supported Values
	RESET		DEFAULT, SYSTEM
DIO	DIR	Set	IN, OUT
		Get	0 to 255 (bit field: 0 = all output, 255 = all input)
	VALUE	Get	0 – 255 (port) 0 – 1 (bit)

Hardware features

- 8 analog input channels, numbered 0 to 7.
  - Gain ranges:
  - ±10V
  - ±5V
  - ±2V
  - ±1V
- 1 digital port.
- All bits are individually configurable as input or output.
- External trigger input
- External pacer input/output
- This feature allows multiple devices on a single USB to acquire synchronized samples. One master device is used to drive the signal. Additional devices must be configured as slave devices using the "AISCAN:EXTPACER=*value*" message. Value may be "ENABLE[/MASTER]", "ENABLE[/SLAVE]" or "DISABLE".
- 1024 bytes of nonvolatile EEPROM memory; used for storing configuration information, calibration data, and user data.
- *RATE* takes a float value.
- If the input scan rate requested is less than the slowest rate supported by the device, the device is set to the slowest rate supported by the device. If the input scan rate requested is greater than the fastest rate supported by the device, an error is generated.

USB-7204

Use the components below to set or get device properties.

Component	Supported Property/Command	Set/Get	Supported Values
AI	CHMODE	Set	SE, DIFF
	OFFSET	Get	4-byte floating point numeric
	SLOPE	Get	4-byte floating point numeric
	VALUE	Get	Counts
AI{ch}	RANGE	Set/Get	BIP20V, BIP10V, BIP5V, BIP4V, BIP2PT5V, BIP2V, BIP1PT25V, BIP1V
AISCAN	EXTPACER	Set/Get	ENABLE/MASTER, ENABLE/SLAVE, ENABLE/GSLAVE
	HIGHCHAN	Set/Get	0 to 7
	LOWCHAN	Set/Get	0 to 7
	QUEUE	Set/Get	ENABLE, DISABLE, RESET
	RATE	Set/Get	0.596 to 50,000 Hz (1 channel)
	RANGE RANGE{ch}	Get	0 - 15 (the number of elements in the queue)
		Set	BIP20V, BIP10V, BIP5V, BIP4V, BIP2PT5V, BIP2V, BIP1PT25V, BIP1V
	RANGE{element/ch}	Set	Element: 0 to 15 Channel: 0 to 7 SE, 0 to 3 DIFF Range: See the range values above.
	SAMPLES	Set/Get	0 to N (0 = continuous scan; N = 32-bit)
	STALL	Set/Get	ENABLE, DISABLE
	START		
	STATUS	Get	IDLE, RUNNING, OVERRUN
	STOP		
	TRIG	Set/Get	ENABLE, DISABLE
	XFRMODE	Set/Get	BLOCKIO, SINGLEIO
AITRIG	REARM	Set/Get	ENABLE, DISABLE
	TYPE	Set/Get	EDGE/RISING, EDGE/FALLING
AO	RANGE	Get	UNI4.096V
	VALUE	Set	0 to 4095
AOSCAN	HIGHCHAN	Set/Get	0 to 1
	LOWCHAN	Set/Get	0 to 1

Component	Supported Property/Command	Set/Get	Supported Values
	RANGE{ch}	Get	UNI4.096V
	RATE	Set/Get	one channel: 1 to 10 kHz two channels: 1 to 5 kHz
	SAMPLES	Set/Get	0 to N (0 = continuous scan; N = 32-bit)
	STALL	Set/Get	ENABLE, DISABLE
	START		
	STATUS	Get	IDLE, RUNNING, UNDERRUN
	STOP		
	TRIG	Set/Get	ENABLE, DISABLE
CTR	START		
	STOP		
	VALUE	Set	0
		Get	0 to 4,294,967,295
DEV	DATATYPE	Set/Get	ENABLE, DISABLE
	FLASHLED	Set	0 to 255
	FWV	Get	MM.mm (M = major; m = minor)
	ID	Set/Get	Up to 57 characters
	MFGCAL	Get	yyyy-mm-dd HH:MM:SS
	MFGCAL{YEAR}	Get	Year as yyyy; 20xx
	MFGCAL{MONTH}	Get	Month as mm; 01 to 12
	MFGCAL{DAY}	Get	Day as dd; 01 to 31
	MFGCAL{HOUR}	Get	Hour as HH; 0 to 23
	MFGCAL{MINUTE}	Get	Minute as MM; 1 to 59
	MFGCAL{SECOND}	Get	Second as SS; 1 to 59
	MFGSER	Get	Up to 8 numeric characters
	RESET		DEFAULT, SYSTEM
DIO	DIR	Set/Get	IN, OUT (port-configurable)
	VALUE	Get	0 to 255 (port) 0 to 1 (bit)

Hardware features

- 8 analog input channels, numbered 0-7.
- Analog input mode is configurable for single-ended (8 channels) or differential (4 channels).
  - Gain ranges:
  - ±20V (differential mode)
  - ±10V (differential or single–ended mode)
  - ±5V (differential mode)
  - ±4V (differential mode)
  - ±2.5V (differential mode)

  - ±2V (differential mode)
  - ±1.25V (differential mode)
  - ±1V (differential mode)
- 2 analog output channels, numbered 0 to 1.
- Analog output range is 0 to 4.096 V, 1 mV per LSB
- 2 digital ports. Each port is individually configurable as input or output.
- External trigger input
- External pacer input/output
- This feature allows multiple devices to acquire synchronized samples. One master device is used to drive the signal. Additional devices must be configured as slave devices using the "AISCAN:EXTPACER=*value*" message. Value may be "ENABLE/MASTER," "ENABLE/SLAVE," or "ENABLE/GSLAVE."
  - When set to *ENABLE/SLAVE*, the first clock pulse after setting up the scan is ignored to ensure adequate setup time for the first conversion. Use this mode when the device is paced from a continuous clock source.
  - When set to *ENABLE/GSLAVE*, the first clock pulse after setting up the scan is held off to ensure adequate setup time for the first conversion. No pulses are ignored. Use this mode when the device is paced from another USB-7204.
- 1024 bytes of nonvolatile EEPROM memory; used for storing configuration information, calibration data, and user data.
- *RATE* takes a float value.
- If the input scan rate requested is less than the slowest rate supported by the device, the device is set to the slowest rate supported by the device. If the input scan rate requested is greater than the fastest rate supported by the device, the device is set to the maximum rate supported by the device.

Analog input data format

Note: This section applies to data acquired using firmware versions ≥2.03. For data acquired using firmware versions ≤2.02, refer to the data format shown below in Table 3.

When performing analog input operations with the "AI" component, the return data is LSB (Least Significant Bit) justified with a value of 0 as Min Scale.

When performing analog input operations with the "AISCAN" component, the differential return data is 12-bit MSB (Most Significant Bit) justified. Additional steps are required to convert single-ended data to 11- or 12-bit representation (LSB justified).

The bit tables below show the data for each operating mode, and lists the steps to convert the data to LSB justified.

Differential mode

Table 1 applies to data acquired using firmware ≥2.03.

Table 1. A/D Converter Values (differential mode)

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	OVR

where:

x The actual conversion value.

OVR 0 = Valid conversion

1 = An analog over-range problem has occurred in the PGA; the conversion value may be invalid. **Note**: this bit can be ignored.

Converting differential data to 12-bit LSB justified

To convert differential data to 12-bit LSB justified, shift the upper 12-bits of data to the right by 4.

The adjusted data format is shown in Analog input data format on page 37.

Single-ended mode

Table 2. A/D Converter Values (single-ended mode)

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data	c	x	x	x	x	x	x	x	x	x	x	x	0	0	0	OVR

where:

c A checkbit for valid A/D transfer functions. When this bit is not set the value is 0.

x The actual conversion value.

OVR 0 = Valid conversion
1 = An analog over-range problem has occurred in the PGA; the conversion value may be invalid. **Note**: this bit can be ignored.

When bit 15 (c) is set or a value greater than 0x7FFF (32,767), you need to convert the data to LSB justified to obtain an 11-bit representation of the 12-bit data.

Converting single-ended data to 12-bit LSB justified

Converting data to LSB justified is required to obtain a 12-bit representation of 11-bit data. Perform the following steps to convert single-ended data to 12-bit LSB justified:

6. Check the value of bit 15 (c) to see if the value is greater than 0x7FFF.
7. If the value is greater than 0x7FFF continue with step 2.
8. If the value is not greater than 0x7FFF then the value is 0 and the conversion procedure is not required.
9. Mask the bits with 0x7FF0.
10. Shift the bits to the right by 3 for 12-bit data, or shift the bits to the right by 4 for 11-bit data.

Table 3 shows the 11-bit data represented as 12-bit data.

Table 3. 11-bit data represented as 12-bit LSB justified (bit 15 is set)

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data	0	0	0	0	x	x	x	x	x	x	x	x	x	x	x	0

where x is the actual conversion value.

Known issues

Firmware version	**Description**
≤2.02	Data overruns may occur or data integrity may be compromised when using AINSCAN to acquire data at rates above 37 kS/s. To resolve this issue, update the device firmware version to 2.03 or later. Refer to the Updating device firmware topic for instructions.

Measurement Computing Corporation
10 Commerce Way
Suite 1008
Norton, Massachusetts 02766
(508) 946-5100
Fax: (508) 946-9500
E-mail: info@mccdaq.com
www.mccdaq.com