

# ADC1000-USB Data Sheet

## Description

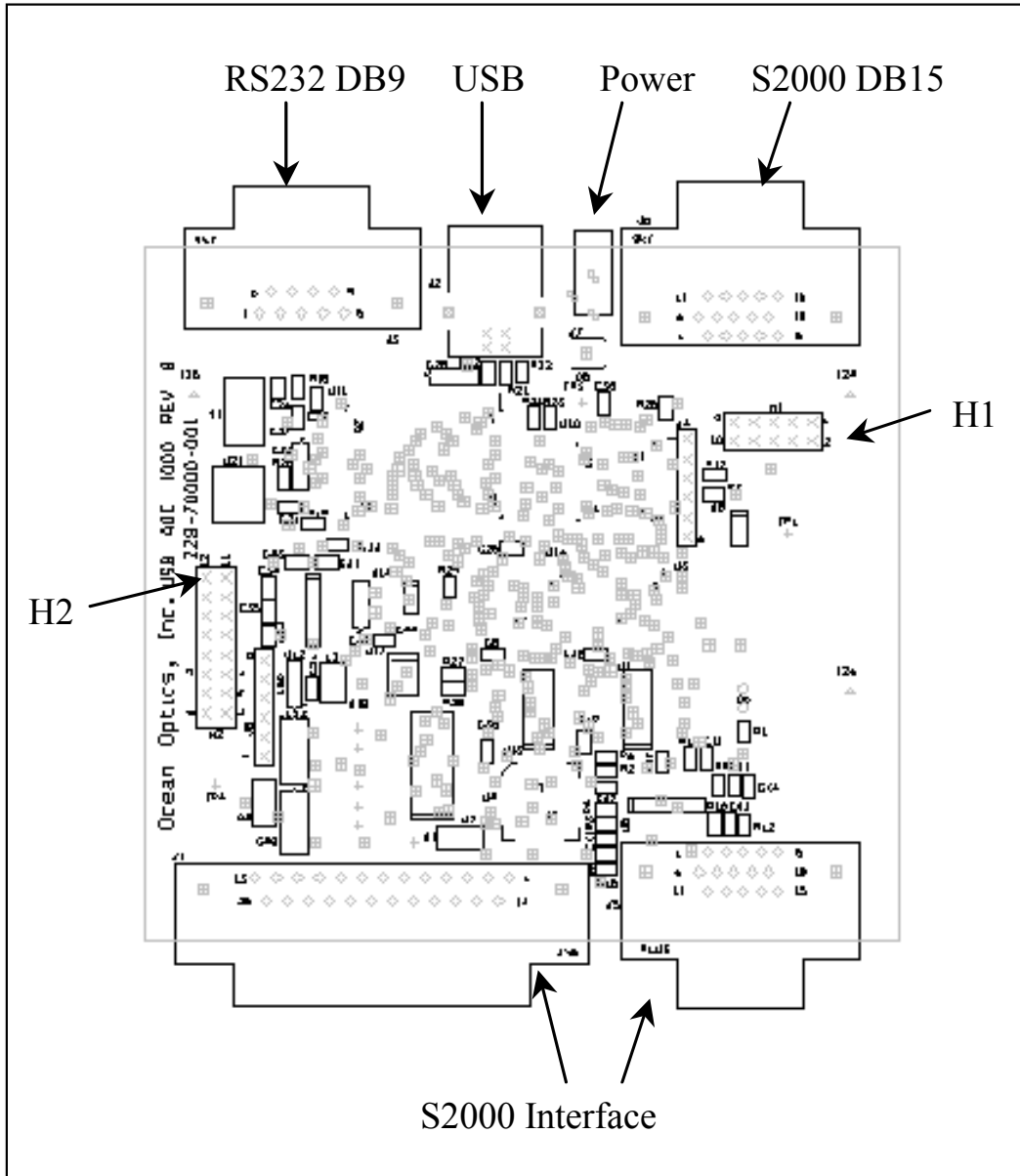
The ADC1000-USB is a microcontroller-based A/D card that can communicate via the Universal Serial Bus or RS-232. This document contains the necessary command information for controlling the unit via the RS-232 or USB interface.

## Features

Listed below are the design specifications for the USB A/D card:

- ❑ PC interface: USB (12Mbps) type B connector
  - Additional RS232 interface capability
- ❑ Supports standard A/D functionality
  - 1MHz A/D rate → 3ms minimum integration time (may be able to run at 2MHz)
  - Controls up to 8 S2000s
  - 8 channel rotator
  - 4 Trigger modes
  - Software controllable continuous strobe rate
  - Strobe enable
  - Low noise
    - Dark Spectra: 3 counts rms
    - S2000 S/N: 260:1
    - Bandwidth: sufficient to preserve a 1 pixel peak
- ❑ S2000 interface options:
  - Plugs directly into DB25 & DB15 of S2000 master
  - Stacks onto the S2000 stack through H1 & H2
- ❑ USB interface can supply power to ~4 S2000s. External power will be required for additional units
- ❑ Software Driver integrated into standard Ocean Optics 32-bit driver library
- ❑ Physical Size: approximately 4" x 4" x 1"

# Mechanical Diagram



Layout of ADC1000-USB

# Hardware Description

The ADC1000-USB uses a Cypress AN2131 microcontroller, which has a high speed 8051, combined with an USB ASIC. Program code and data coefficients are stored in external EEPROM, which are loaded at boot-up via the I<sup>2</sup>C bus. The microcontroller has 8K of internal SRAM and 128K of external SRAM. Due to memory mapping and paging constraints, only 96K of the external SRAM is useable.

The ADC1000-USB can be controlled either through the serial or USB interface. The following sections provide the command set for each interface

## USB Information

Ocean Optics Vendor ID number is 0x2457. The ADC1000-USB can have 2 Product IDs depending upon the EEPROM configuration. In the case where the code is loaded from the EEPROM the PID is 0x1004. The AN2131 allows for the code to be loaded from the host processor (Re-numeration), in this case the PID is 0x1003.

## Serial Instruction Set

### Serial Command Syntax

The list of the commands are shown in the following table along with the microcode version number they were introduced with. All commands consist of an ASCII character passed over the serial port, followed by some data. The length of the data depends on the command. The format for the data is either ASCII or binary (default). The ASCII mode is set with the “a” command and the binary mode with the “b” command. To insure accurate communications, all commands respond with an ACK (ASCII 6) for an acceptable command or a NAK (ASCII 21) for an unacceptable command (i.e. data value specified out of range).

In the ASCII data value mode, the ADC1000-USB “echoes” the command back out the RS-232 port. In binary mode all data, except where noted, passes as 16-bit unsigned integers (WORDS) with the MSB followed by the LSB. By issuing the “v command” (Version number query), the data mode can be determined by viewing the response (ASCII or binary).

In a typical data acquisition session, the user sends commands to implement the desired spectral acquisition parameters (integration time, A/D Channel, etc.). Then the user sends commands to acquire spectra (S command) with the previously set parameters. If necessary, the baud rate can be changed at the beginning of this sequence to speed up the data transmission process.

### Serial Command Summary

Letter	Description	Version
A	Adds scans	1.00.0
B	Set Pixel Boxcar	1.00.0
C		
D		
E		
F	Non functional but follows SAD500 command format*	1.00.0
G	Set Data Compression	1.00.0
H	Set A/D Channel	1.00.0
I	Sets integration time	1.00.0
J	Sets Lamp Enable Line	1.00.0
K	Changes baud rate	1.00.0
L		
M		
N		
O		
P	Partial Pixel Mode	1.00.0
Q	Initializes operating values	1.00.0
R		
S	Starts spectral acquisition with previously set parameters	1.00.0
T	Sets trigger mode	1.00.0
U		
V		
W		
X		
Y		
Z		
a	Set ASCII mode for data values	1.00.0
b	Set binary mode for data values	1.00.0
f	Set Continuous Strobe Rate	1.00.0
k	Sets Checksum mode	1.00.0
t		
v	Provides microcode version #	1.00.0

Letter	Description	Version
x	Sets calibration coefficients	1.00.0
?	Queries parameter values	1.00.0
-	ADC1000-USB Identifier	1.00.0
*"Non functional but follows SAD500 command format" means that the ADC1000-USB will require a data value following the command variable and respond with just one NAK (instead of 3). This allows customer device drivers that were written for the SAD500 to be functional with minimal change		

## Command Descriptions

A detailed description of all ADC1000-USB commands follows. The {} indicates a data value which is interpreted as either ASCII or binary (default). The default value indicates the value of the parameter upon power up.

### Add Scans

Description: Sets the number of discrete spectra to be summed together. Since this routine can add up to 15 spectra, each with a maximum intensity of 4096, the maximum returned intensity is 65535.

Command Syntax:	A{DATA WORD}
Response:	ACK or NAK
Range:	1-15
Default value:	1

### Pixel Boxcar Width

Description: Sets the number of pixels to be averaged together. A value of  $n$  specifies the averaging of  $n$  pixels to the right and  $n$  pixels to the left. This routine uses 32-bit integers so that intermediate overflow will not occur; however, the result is truncated to a 16-bit integer prior to transmission of the data. This math is performed just prior to each pixel value being transmitted out. Values greater than  $\sim 3$  will exceed the idle time between values and slow down the overall transfer process.

Command Syntax:	B{DATA WORD}
Response:	ACK or NAK
Range:	0-15
Default value:	0

## Set Data Compression

Description: Specifies whether the data transmitted from the ADC1000-USB should be compressed to speed data transfer rates. For more information on ADC1000-USB Data Compression, see [Technical Note 1: ADC1000-USB Data Compression](#).

Command Syntax:	G{DATA WORD}
Response:	ACK or NAK
Range:	0 – Compression off !0 – Compression on
Default value:	0

## Spectrometer Channel

Description: Sets the spectrometer channel to be digitized. The data value will either activate or deactivate the rotator feature. If the rotator is active, then the pixels are interlaced in the following manner

Pixel	0	1	...	N	N+1	N+2	...	2N	...
Channel	Master	Slave1	...	Slave N	Master	Slave 1	...	Slave N...	

Command Syntax:	H{DATA WORD}
Response:	ACK or NAK
Range:	0-7, 256 – 263 If the value is > 255 then the rotator is active, otherwise it is inactive
Default value:	0

## Integration Time

Description: Sets the ADC1000-USBs integration time, in milliseconds, to the value specified.

Command Syntax:	I{DATA WORD}
Response:	ACK or NAK
Range:	5 - 65535
Default value:	100

If the Integration clock is using the 8-bit timer (y command) and a value greater than 255 is desired, the integration time is set to the LSB of the data word (i.e., value MOD 255).

## Lamp Enable

Description: Sets the ADC1000-USB's Lamp Enable line to the value specified

Command Syntax:	J{DATA WORD}
Value:	0 = Light source/strobe off—Lamp Enable low !0 = Light source/strobe on—Lamp Enable high
Response:	ACK or NAK
Default value:	0

## Baud Rate

Description: Sets the ADC1000-USB's baud rate.

Command Syntax:	K{DATA WORD}
Value:	0=2400 1=4800 2=9600 3=19200 4=38400 5=57600 6=115200
Response:	See below
Default value:	2

When changing baud rates, this sequence must be followed:

1. Controlling program sends K with desired baud rate, communicating at the old baud rate
2. A/D responds with ACK at old baud rate, otherwise it responds with NAK and the process is aborted
3. Controlling program waits longer than 50 milliseconds
4. Controlling program sends K with desired baud rate, communicating at the new baud rate
5. A/D responds with ACK at new baud rate, otherwise it responds with NAK and old baud rate is used.

---

### Note

If a deviation occurs at any step, the previous baud rate is used.

---

## Pixel Mode

Description: Specifies which pixels are transmitted. While all pixels are acquired on every scan, this parameter determines which pixels will be transmitted out the serial port.

### Note

Since most applications only require a subset of the spectrum, this mode can greatly reduce the amount of time required to transmit a spectrum while still providing all of the desired data. This mode is helpful when interfacing to PLCs or other processing equipment.

Command Syntax:	P{DATA WORD}	
Value:	Description 0 = all 2048 pixels 1 = every $n^{\text{th}}$ pixel with no averaging 2 = N/A 3 = pixel x through y every n pixels 4 = up to 10 randomly selected pixels between 0 and 2047 (denoted p1, p2, ... p10)	Example P 0 (spaces for clarity only) P 1 n P 2 n  P 3 x y n P 4 n p1 p2 p3...p10
Response:	ACK or NAK	
Default value:	0	

## Initialize

Description: Sets all operating parameters to their default value.

Command Syntax:	Q
Response:	ACK or NAK

## Spectral Acquisition

Description: Acquires spectra with the current set of operating parameters. When executed, this command determines the amount of memory required. If sufficient memory does not exist, an ETX (ASCII 3) is immediately returned and no spectra are acquired, otherwise an STX (ASCII 2) is returned followed by the data.

Command Syntax:	S
Response:	If successful, STX followed by data If unsuccessful, ETX

The format of returned spectra includes a header to indicate scan number, channel number, pixel mode, etc. The format is as follows:

- WORD 0xFFFF – start of spectrum
- WORD A/D channel number
- WORD scan number ALWAYS 0
- WORD scans in memory ALWAYS 0
- WORD integration time in milliseconds
- WORD integration time counter ALWAYS 0
- WORD pixel mode
- WORDS if pixel mode not 0, indicates parameters passed to the Pixel Mode command (P)
- WORDS spectral data
- WORD 0xFFFD – end of spectrum

### Trigger Mode

Description: Sets the ADC1000-USB's external trigger mode to the value specified.

Command Syntax:	T{DATA WORD}
Value:	0 = Normal – Continuously scanning 1 = Software trigger 2 = External Synchronization 3 = External Hardware Trigger
Response:	ACK or NAK
Default value:	0

### ASCII Data Mode

Description: Sets the mode in which data values are interpreted to be ASCII. Only unsigned integer values (0 – 65535) are allowed in this mode and the data values are terminated with a carriage return (ASCII 13) or linefeed (ASCII 10). In this mode the ADC1000-USB “echoes” the command and data values back out the RS-232 port.

---

### Notes

The command requires that the string “aA” be sent without any CR or LF. This is an attempt to insure that this mode is not entered inadvertently.

A legible response to the Version number query (v command) indicates the unit is in the ASCII data mode.

---

Command Syntax:	aA
Response:	ACK or NAK
Default value	N/A

## Binary Data Mode

Description: Sets the mode in which data values are interpreted to be binary. Only 16 bit unsigned integer values (0 – 65535) are allowed in this mode with the MSB followed by the LSB.

### Note

The command requires that the string “bB” be sent without any CR or LF. This is an attempt to insure that this mode is not entered inadvertently.

Command Syntax:	bB
Response:	ACK or NAK
Default value	Default at power up – not changed by Q command

## Set Continuous Strobe Rate

Description: Sets the continuous strobe period, in milliseconds, to the value specified. A value greater than 255 results in the strobe rate being set to 255.

Command Syntax:	f{DATA WORD}
Response:	ACK or NAK
Range:	1 - 255
Default value:	10

## Checksum Mode

Description: Specifies whether the ADC1000-USB will generate and transmit a 16-bit checksum of the spectral data. This checksum can be used to test the validity of the spectral data, and its use is recommended when reliable data scans are required. See [Technical Note 2: ADC1000-USB Checksum Calculation](#) for more information on checksum calculation.

Command Syntax:	k{DATA WORD}
Value:	0 = Do not transmit checksum value !0 = transmit checksum value at end of scan
Response:	ACK or NAK
Default value:	0

**Version Number Query**

Description: Returns the version number of the code running on the microcontroller. A returned value of 1000 is interpreted as 1.00.0.

Command Syntax:	v
Response:	ACK followed by {DATA WORD}
Default value	N/A

**Calibration Constants**

Description: Writes one of the 44 possible calibration constant to EEPROM. The calibration constant is specified by the first DATA WORD which follows the x. The calibration constant is stored as an ASCII string with a max length of 15 characters. The string is not check to see if it makes sense.

Command Syntax:	x{DATA WORD}{ASCII STRING}
Value:	<p>DATA WORD Index description:</p> <ul style="list-style-type: none"> <li>0 – Serial Number</li> <li>1 – Channel Enabled Register – bit n, channel n, ...</li> <li>2 – 0<sup>th</sup> order Wavelength Calibration Coefficient: Channel 0</li> <li>3 – 1<sup>st</sup> order Wavelength Calibration Coefficient: Channel 0</li> <li>4 – 2<sup>nd</sup> order Wavelength Calibration Coefficient: Channel 0</li> <li>5 – 3<sup>rd</sup> order Wavelength Calibration Coefficient: Channel 0</li> <li>6 – 0<sup>th</sup> order Wavelength Calibration Coefficient: Channel 1</li> <li>7 – 1<sup>st</sup> order Wavelength Calibration Coefficient: Channel 1</li> <li>8 – 2<sup>nd</sup> order Wavelength Calibration Coefficient: Channel 1</li> <li>9 – 3<sup>rd</sup> order Wavelength Calibration Coefficient: Channel 1</li> <li>...</li> <li>30 – 0<sup>th</sup> order Wavelength Calibration Coefficient: Channel 7</li> <li>31– 1<sup>st</sup> order Wavelength Calibration Coefficient: Channel 7</li> <li>32– 2<sup>nd</sup> order Wavelength Calibration Coefficient: Channel 7</li> <li>33 – 3<sup>rd</sup> order Wavelength Calibration Coefficient: Channel 7</li> <li>34 - 44 – Reserved</li> </ul>
Response:	ACK or NAK
Default value:	N/A

To query the constants, use the ?x{DATA WORD} format to specify the desired constant.

## Query Variable

Description: Returns the current value of the parameter specified. The syntax of this command requires two ASCII characters. The second ASCII character corresponds to the command character which sets the parameter of interest (acceptable values are B, A, I, K, T, J). A special case of this command is ?x (lower case) which requires an additional data word be passed to indicate which calibration constant is to be queried.

Command Syntax:	?{ASCII character}
Response:	ACK followed by {DATA WORD}
Default value:	N/A

## ADC1000-USB Identifier

Description: Device drivers can differentiate between an ADC1000-USB, ADC1000-USB and SAD500 with the use of this command. The ADC1000-USB will generate an ACK in response to this command while the other will respond with a NAK.

Command Syntax:	-
Response:	ACK
Default value	N/A

## Examples

Below are examples on how to use some of the commands. Commands are in **BOLD** and descriptions are in parenthesis. For clarity, the commands are shown in the ASCII mode (a command) instead of the default binary mode.

The desired operating conditions are: acquire spectra from spectrometer channel 0 (master) with a 200ms integration time, set number of scan to add to 5 and operate at 57,600 Baud.

```

aA      (Set ASCII Data Mode)
K5<CR>   (Start baud rate change to 57,600)
           Wait for ACK, change to 57600, wait for 50ms
K5<CR>   (Verify command, communicate at 57600)
A5<CR>   (Add 5 spectra)
I200<CR> (Set integration time to 200ms)
S       (Acquire spectra)
...       Repeat as necessary

```

## Application Tips

During the software development phase of a project, the operating parameters of the ADC1000-USB may become out-of-synch with the controlling program. It is good practice to cycle power on the ADC1000-USB when errors occur.

If you question the state of the ADC1000-USB, you can transmit a space (or another non-command) using a terminal emulator. If you receive a NAK, the ADC1000-USB is awaiting a command; otherwise, it is still completing the previous command.

For Windows 95/98 users, use HyperTerminal as a terminal emulator after selecting the following:

1. Select File | Properties.
2. Under Connect using, select Direct to Com x.
3. Click Configure and match the following Port Settings:
  - Bits per second (Baud rate): Set to desired rate
  - Data bits: 8
  - Parity: None
  - Stop bits: 1
  - Flow control: None
4. Click the **Advanced** button and slide the **Receive Buffer** and **Transmit Buffer** arrows to the Low 1 value. Click **OK**.
5. Click **OK** in **Port Settings** and in **Properties** dialog boxes.

## USB Instruction Set

### USB Command Syntax

The list of the commands is shown in the following table followed by a detailed description of each command. The length of the data depends on the command. All commands are sent to the ADC1000-USB through End Point 2 (EP2). All spectra data is acquired through End Point 2 In and all other queries are retrieved through End Point 7 In (EP7).

Pipe #	Description	Type	Full Speed Size (Bytes)	Endpoint Address
0	End Point 2 Out	Bulk	64	0x02
1	End Point 2 In	Bulk	64	0x82
2	End Point 7 Out (unused)	Bulk	64	0x07
3	End Point 7 In	Bulk	64	0x87

### USB Command Summary

EP2 Command Byte Value	Description	Version
0x01	Reset ADC1000-USB	1.00.0
0x02	Set Integration Time	1.00.0
0x03	Set Strobe Enable Status	1.00.0
0x04	Reserved	1.00.0

EP2 Command Byte Value	Description	Version
0x05	Query Information	1.00.0
0x06	Write Information	1.00.0
0x07	Write Serial Number	1.00.0
0x08	Get Serial Number	1.00.0
0x09	Request Spectra	1.00.0
0x0A	Set Trigger Mode	1.00.0
0x0B	Set Spectrometer Channel	1.00.0
0x0C	Set Continuous Strobe Rate	1.00.0
0x0D	Set Master Clock Rate	1.00.0

## USB Command Descriptions

A detailed description of all ADC1000-USB commands follows. While all commands are sent to EP2 over the USB port, the byte sequence is command dependent. The general format is the first byte is the command value and the additional bytes are command specific values.

Byte 0	Byte 1	Byte 2	...	Byte n-1
Command Byte	Command Specific	Command Specific	...	Command Specific

### Reset ADC1000-USB

Description: This command should be called at the start of every session. It resets the device and acquires a spectrum. The application needs to get this spectral data to maintain effective communications (refer to the spectral acquisition command for details).

#### Byte Format

Byte 0
0x01

### Set Integration Time

Description: Sets the ADC1000-USB integration time in milliseconds. The acceptable range is 3 – 65535. If the value is less than 3ms then the integration time is unchanged.

#### Byte Format

Byte 0	Byte 1	Byte 2
0x02	Integration Time LSB	Integration Time MSB

### Set Strobe Enable Status

Description: Sets the ADC1000-USB S0 signal (Strobe Enable) line as follows.

Data Byte = 0 → Lamp Enable Low/Off  
 Data Byte = !0 → Lamp Enable HIGH/On

**Byte Format**

Byte 0	Byte 1	Byte 2
0x03	Data byte LSB	Data Byte MSB

### Query Information

Description: Queries any of the 45 stored spectrometer configuration variables. The Query command is sent to EP2 and the data is retrieved through End Point 7. The 45 configuration variables are indexed as follows:

Data Byte - Description

- 0 – Serial Number
- 1 – Channel Enabled Register – bit n, channel n, ...
- 2 – 0<sup>th</sup> order Wavelength Calibration Coefficient: Channel 0
- 3 – 1<sup>st</sup> order Wavelength Calibration Coefficient: Channel 0
- 4 – 2<sup>nd</sup> order Wavelength Calibration Coefficient: Channel 0
- 5 – 3<sup>rd</sup> order Wavelength Calibration Coefficient: Channel 0
- 6 – 0<sup>th</sup> order Wavelength Calibration Coefficient: Channel 1
- 7 – 1<sup>st</sup> order Wavelength Calibration Coefficient: Channel 1
- 8 – 2<sup>nd</sup> order Wavelength Calibration Coefficient: Channel 1
- 9 – 3<sup>rd</sup> order Wavelength Calibration Coefficient: Channel 1
- ...
- 30 – 0<sup>th</sup> order Wavelength Calibration Coefficient: Channel 7
- 31 – 1<sup>st</sup> order Wavelength Calibration Coefficient: Channel 7
- 32 – 2<sup>nd</sup> order Wavelength Calibration Coefficient: Channel 7
- 33 – 3<sup>rd</sup> order Wavelength Calibration Coefficient: Channel 7
- 34 - 44 – Reserved

**Byte Format**

Byte 0	Byte 1
0x05	Data byte

### Return Format (EP7)

The data is returned in ASCII format and read in by the host through End Point 7.

Byte 0	Byte 1	Byte 2	Byte 3	...	Byte 16
0x05	Configuration Index	ASCII byte 0	ASCII byte 1	...	ASCII byte 14

### Write Information

Description: Writes any of the 45 stored spectrometer configuration variables to EEPROM. The 45 configuration variables are indexed as described in the Query Information. The information to be written is transferred as ASCII information. The time required to complete this command is ~150ms. The controlling program should allow sufficient time in between these operations.

#### Byte Format

Byte 0	Byte 1	Byte 2	Byte 3	...	Byte 16
0x06	Configuration Index	ASCII byte 0	ASCII byte 1	...	ASCII byte 14

### Write Serial Number

Description: Writes the serial number to EEPROM. The information to be written is transferred as ASCII information. The time required to complete this command is ~150ms.

#### Byte Format

Byte 0	Byte 1	Byte 2	...	Byte 16
0x07	ASCII byte 0	ASCII byte 1	...	ASCII byte 15

### Query Serial Number

Description: Queries the unit's serial number. The Query command is sent to EP2 and the data is retrieved through End Point 7. The information to be read is transferred as ASCII information.

#### Byte Format

Byte 0
0x08

**Return Format**

The data is returned in ASCII format and read in by the host through End Point 7.

<b>Byte 0</b>	<b>Byte 1</b>	<b>Byte 2</b>	...	<b>Byte 16</b>
0x08	ASCII byte 0	ASCII byte 1	...	ASCII byte 15

**Request Spectra**

Description: Initiates a spectra acquisition. The ADC1000-USB will acquire a complete spectrum (2048 pixel values) for the current operating parameters. The data is returned in bulk transfer mode through EP2 in 64 packets each containing 64 bytes. The pixel values are decoded as described below.

**Byte Format**

<b>Byte 0</b>
0x09

**Return Format**

The data is returned in bulk transfer mode through EP2 in 64 packets each containing 64 bytes. There is an additional packet containing one value that is used as a flag to insure proper synchronization between the PC and ADC1000-USB. The pixel values are decoded as described below.

**Packet 0 – LSBs for first 64 pixels**

<b>Byte 0</b>	<b>Byte 1</b>	...	<b>Byte 63</b>
Pixel 0 LSB	Pixel 1 LSB	...	Pixel 63 LSB

**Packet 1 – MSBs for first 64 pixels**

<b>Byte 0</b>	<b>Byte 1</b>	...	<b>Byte 63</b>
Pixel 0 MSB	Pixel 1 MSB	...	Pixel 63 MSB

**Packet 2 – LSBs for 2<sup>nd</sup> group of 64 pixels**

<b>Byte 0</b>	<b>Byte 1</b>	...	<b>Byte 63</b>
Pixel 64 LSB	Pixel 65 LSB	...	Pixel 127 LSB

**Packet 3 – MSBs for 2<sup>nd</sup> group of 64 pixels**

<b>Byte 0</b>	<b>Byte 1</b>	...	<b>Byte 63</b>
Pixel 64 MSB	Pixel 65 MSB	...	Pixel 127 MSB

...

**Packet 62 – LSBs for last group of 64 pixels**

<b>Byte 0</b>	<b>Byte 1</b>	...	<b>Byte 63</b>
Pixel 1984 LSB	Pixel 1985 LSB	...	Pixel 2047 LSB

**Packet 63 – MSBs for last group of 64 pixels**

<b>Byte 0</b>	<b>Byte 1</b>	...	<b>Byte 63</b>
Pixel 1984 MSB	Pixel 1985 MSB	...	Pixel 2047 MSB

**Packet 64 – Synchronization Packet**

<b>Byte 0</b>
0x69

**Set Trigger Mode**

Description: Sets the ADC1000-USB Trigger mode to one of four states. If an unacceptable value is passed then the trigger state is unchanged.

Data Value = 0 → Normal (Free running) Mode
Data Value = 1 → Software Trigger Mode
Data Value = 2 → External Synchronization
Data Value = 3 → External Hardware Trigger Mode

**Byte Format**

<b>Byte 0</b>	<b>Byte 1</b>	<b>Byte 2</b>
0x0A	Data Value LSB	Data Value MSB

### Set Spectrometer Channel

Description: Sets the ADC1000-USB spectrometer channel to the desired S2000 unit. If the 8<sup>th</sup> bit is set the rotator feature is enabled, otherwise it is disabled. The rotator allows acquisition from multiple spectrometers on the same integration period. When this feature is enabled, the pixel resolution of each spectrometer channel is reduced by the number of channels rotated through. For example, if the rotator was enabled with a value of 0x103 to allow rotation through channels 0 - 3, then the ADC1000-USB would acquire pixel data in the following order:

0, 1, 2, 3, 0, 1, 2, 3, ...                      or  
M, S1, S2, S3, M, S1, S2, S3    in master and slave channel notation

Data Value = 0x0000 -0007 → Set MUX to Spectrometer channel 0-7, disable rotator
Data Value = 0x101 → Enable rotation up through channel 1
Data Value = 0x102 → Enable rotation up through channel 2
Data Value = 0x103 → Enable rotation up through channel 3
...
Data Value = 0x107 → Enable rotation up through channel 7

#### Byte Format

Byte 0	Byte 1	Byte 2
0x0B	Data Value LSB	Data Value MSB

### Set Continuous Strobe Rate

Description: Sets the Continuous Strobe Signal (H2: pin 10) to the desired period in milliseconds. Acceptable values are 0ms to 255ms. Values greater than 255ms result in the strobe rate being set to 255ms. A value of 0 will disable the strobe at its current state (either HIGH or LOW).

#### Byte Format

Byte 0	Byte 1	Byte 2
0x0C	Continuous Strobe LSB	Continuous Strobe MSB

### Set Master Clock Frequency

Description: Sets the A/D conversion Frequency to the desired value in KHz. The Master Clock signal (H2: pin 12) is driven to twice this frequency to meet the S2000 requirement in which the A/D conversion rate is half the master clock rate. Acceptable values are 8 to 1000KHz. The master clock signal is derived with an 8-bit counter running at a max frequency of 2MHz. Values less than 8KHz result in the master clock rate being set to 8KHz.

### Byte Format

Byte 0	Byte 1	Byte 2
0x0D	Master Clock LSB	Master Clock MSB

## Technical Note 1: ADC1000-USB Data Compression

Transmission of spectral data over the serial port is a relatively slow process. Even at 57,600 baud, the transmission of a complete 2048 point spectrum takes around 600 msec. The ADC1000-USB implements a data compression routine that minimizes the amount of data that needs to be transferred over the RS-232 connection. Using the “G” command (Compressed Mode) and passing it a parameter of 1 enables the data compression. Every scan transmitted by the ADC1000-USB will then be compressed. The compression algorithm is as follows:

1. The first pixel (a 16-bit unsigned integer) is always transmitted uncompressed.
2. The next byte is compared to 0x80.
3. If the byte is equal to 0x80, the next two bytes are taken as the pixel value (16-bit **unsigned** integer).
4. If the byte is not equal to 0x80, the value of this byte is taken as the difference in intensity from the previous pixel. This difference is interpreted as an 8-bit **signed** integer.
5. Repeat step 2 until all pixels have been read.

Using this data compression algorithm greatly increases the data transfer speed of the ADC1000-USB. The table below shows the data transfer speed, in milliseconds, for various light sources and baud rates. Keep in mind that these rates are for demonstration purposes only, and the speed of your computer may impact the data transfer rates.

	Comp	57.6 kb	% faster	38.4 kb	% faster	19.2 kb	% faster	9.6 kb	% faster
dark	on	426	45.2%	624	46.7%	1148	47.5%	2247	48.8%
	off	778		1170		2188		4391	
LS-1	on	429	44.9%	624	46.6%	1141	49.6%	2192	50.1%
	off	779		1169		2266		4390	
HG-1	on	465	40.2%	679	41.9%	1238	43.5%	2424	44.8%
	off	777		1169		2193		4391	

The following table shows a section of a spectral line source spectrum and the results of the data compression algorithm.

<b>Pixel Value</b>	<b>Value Difference</b>	<b>Transmitted Bytes</b>
185	0	0x80 0x00 0xB9
2151	1966	0x80 0x08 0x67
836	-1315	0x80 0x03 0x44
453	-383	0x80 0x01 0xC5
210	-243	0x80 0x00 0xD2
118	-92	0xA4
90	-28	0xE4
89	-1	0xFF
87	-2	0xFE
89	2	0x02
86	-3	0xFD
88	2	0x02
98	10	0x0A
121	23	0x17
383	262	0x80 0x01 0x7F
1162	779	0x80 0x04 0x8A
634	-528	0x80 0x02 0x7A
356	-278	0x80 0x01 0x64
211	-145	0x80 0x00 0xD3
132	-79	0xB1
88	-44	0xD4
83	-5	0xFB
86	3	0x03
82	-4	0xFC
91	9	0x09
92	1	0x01
81	-11	0xF5
80	-1	0xFF
84	4	0x04
84	0	0x00
85	1	0x01
83	-2	0xFE
80	-3	0xFD
80	0	0x00
88	8	0x08
94	6	0x06

Pixel Value	Value Difference	Transmitted Bytes
90	-4	0xFC
103	13	0x0D
111	8	0x08
138	27	0x1B

In this example, spectral data for 40 pixels is transmitted using only 60 bytes. If the same data set were transmitted using uncompressed data, it would require 80 bytes.

## Technical Note 2: ADC1000-USB Checksum Calculation

For all uncompressed pixel modes, the checksum is simply the unsigned 16-bit sum (ignoring overflows) of all transmitted spectral points. For example, if the following 10 pixels are transferred, the calculation of the checksum would be as follows:

Pixel Number	Data (decimal)	Data (hex)
0	15	0x000F
1	23	0x0017
2	46	0x002E
3	98	0x0062
4	231	0x00E7
5	509	0x01FD
6	1023	0x03FF
7	2432	0x0980
8	3245	0x0CAD
9	1984	0x07C0

Checksum value: 0x2586

When using a data compression mode, the checksum becomes a bit more complicated. A compressed pixel is treated as a 16-bit **unsigned** integer, with the most significant byte set to 0. Using the same data set used in [Technical Note 1: ADC1000-USB Data Compression](#), the following shows a section of a spectral line source spectrum and the results of the data compression algorithm.

Data Value	Value Difference	Transmitted Bytes	Value Added to Checksum
185	0	0x80 0x00 0xB9	0x0139
2151	1966	0x80 0x08 0x67	0x08E7
836	-1315	0x80 0x03 0x44	0x03C4
453	-383	0x80 0x01 0xC5	0x0245
210	-243	0x80 0x00 0xD2	0x0152
118	-92	0xA4	0x00A4
90	-28	0xE4	0x00E4
89	-1	0xFF	0x00FF
87	-2	0xFE	0x00FE
89	2	0x02	0x0002
86	-3	0xFD	0x00FD
88	2	0x02	0x0002
98	10	0x0A	0x000A
121	23	0x17	0x0017
383	262	0x80 0x01 0x7F	0x01FF
1162	779	0x80 0x04 0x8A	0x050A
634	-528	0x80 0x02 0x7A	0x02FA
356	-278	0x80 0x01 0x64	0x01E4
211	-145	0x80 0x00 0xD3	0x0153
132	-79	0xB1	0x00B1
88	-44	0xD4	0x00D4
83	-5	0xFB	0x00FB
86	3	0x03	0x0003
82	-4	0xFC	0x00FC
91	9	0x09	0x0009
92	1	0x01	0x0001
81	-11	0xF5	0x00F5
80	-1	0xFF	0x00FF
84	4	0x04	0x0004
84	0	0x00	0x0000
85	1	0x01	0x0001
83	-2	0xFE	0x00FE
80	-3	0xFD	0x00FD
80	0	0x00	0x0000
88	8	0x08	0x0008

**ADC1000-USB Data Sheet**

<b>Data Value</b>	<b>Value Difference</b>	<b>Transmitted Bytes</b>	<b>Value Added to Checksum</b>
94	6	0x06	0x0006
90	-4	0xFC	0x00FC
103	13	0x0D	0x000D
111	8	0x08	0x0008
138	27	0x1B	0x001B

The checksum value is simply the sum of all entries in the last column, and evaluates to 0x2C13.