

# Jaz Data Sheet

## Description

Jaz is a community of stackable, modular and autonomous component modules that combine to create a community of smart sensing instruments that is unfettered by the limits of traditional optical sensing instrumentation: a powerful microprocessor and onboard display eliminate the need for a PC; stackable, autonomous instrument modules allow users to customize the system to their changing application needs; and Ethernet connectivity plus SD card data storage capability facilitate remote operation.

Jaz is built on a platform that expands to include one or more light sources (UV-VIS, VIS-NIR, PX, LED) rechargeable lithium-ion battery and up to 8 spectrometer modules to make Jaz the first analytical instrument easily adaptable for the field, lab or process environment. Jaz can also connect to a computer via the USB port. When connected through a USB 2.0 or 1.1, the spectrometer draws power from the host computer. Jaz, like all USB devices, can be controlled by our SpectraSuite software, a completely modular, Java-based spectroscopy software platform that operates on Windows, Macintosh and Linux operating systems.

Jaz is extremely adaptable. It can be powered via battery, USB connection, over the Ethernet (using Power over Ethernet, a special feature on some network infrastructure devices) , or from a power outlet. The Jaz display can be rotated 180 degrees in either direction and the faceplate itself can be specified for either orientation. You can order the specific modules needed for your particular application to be built in the Jaz stack, making Jaz a uniquely customizable scientific instrument and eliminating the cost of unnecessary components.



The detector used in the Jaz spectrometer is a high-sensitivity 2048-element CCD array from Sony, product number ILX511B. (For complete details on this detector, visit Sony's web site at [www.sony.com](http://www.sony.com). Ocean Optics applies a coating to all ILX511B detectors, so the optical sensitivity could vary from that specified in the Sony datasheet.)

## Jaz Modules

Jaz consists of a stack of individual spectroscopic instruments (currently up to 8 devices in a stack) specified by the user to suit their unique application. Modules available for the Jaz stack include the following:

Module	Description
JAZ-S*	Spectrometer and grating
JAZ-DPU	Processor (CPU), OLED display and keypad
JAZ-E	Ethernet and memory module. Includes one slot for an SD card (up to 2 GB).
JAZ-B	Battery (lithium-ion) and memory module. Includes two slots for SD cards (up to 2 GB each).
JAZ-PX	Pulsed Xenon light source
JAZ-UV-VIS	Deuterium-tungsten halogen light source (maximum 1 if stack is powered by a battery, or more if each is individually powered)
JAZ-VIS-NIR	Tungsten-halogen light source
JAZ-L640*	640 nm red LED
JAZ-L590*	590 nm yellow LED
JAZ-L450*	450 nm blue LED
JAZ-L405*	405 nm LED
JAZ-L365	365 nm UV LED
JAZ-LWHITE*	White LED
JAZ-NEOFOX	NeoFox Sport oxygen sensing module for measuring oxygen partial pressure and dissolved oxygen. Maximum one per stack.
JAZ-INDY	Industrial module for RS-232 and RS-485 communication. Also offers analog and digital outputs.
* Maximum 8 per stack	

## Features

### Spectrometer Module

- ❑ Sony ILX511B 2048-element linear silicon CCD array detector
- ❑ 14 grating options, UV through shortwave NIR
- ❑ Responsive from 200 to 1100 nm
- ❑ Maximum wavelength range 650 nm

- ❑ Sensitivity of up to 75 photons/count at 400 nm; 41 photons/count at 600 nm
- ❑ An optical resolution of ~0.3 to 10.0 nm (FWHM)
- ❑ Integration times from 1 ms to 65 seconds
- ❑ 16 bit, 3 MHz A/D Converter
- ❑ 4 triggering modes
- ❑ Embedded microcontroller allows programmatic control of all operating parameters and standalone operation
- ❑ EEPROM storage for
  - Wavelength Calibration Coefficients
  - Linearity Correction Coefficients

#### **Ethernet and Battery Modules**

- ❑ SD card data storage
- ❑ 8-hour rechargeable lithium-ion battery for JAZ-COMBO

#### **Jaz Stack**

- ❑ Low power consumption of ~ 2.5 W (JAZ-COMBO)
- ❑ 19-pin connector for interfacing to external products
- ❑ Onboard GPIO – 4 user-programmable digital I/Os
- ❑ Up to 8 spectrometers in a stack for simultaneous multipoint measurement

## **Specifications**

Specification	Value
Power Consumption	500 mA @ 5 VDC for JAZ-COMBO
Computer interface	USB in JAZ-DPU module Ethernet in optional JAZ-E module RS-232 in JAZ-DPU and JAZ-INDY modules RS-485 in JAZ-INDY module
Computer requirements (if connected to Jaz)	IBM-compatible PC with Pentium or better microprocessor 512 MB RAM Windows XP, Mac OS X and Linux when using the USB interface on desktop or notebook PCs.
Available operating software	SpectraSuite (for an additional cost)
Fiber optic connector	SMA 905 to single-strand optical fiber
<b>Spectrometer Module (JAZ-S)</b>	
Grating Options	14 different gratings, UV through Shortwave NIR
Detector	2048-element linear silicon CCD array; L2 Collection Lens (optional)
Wavelength range	200-1100 nm; maximum 650 nm
Entrance aperture	5, 10, 25, 50, 100 or 200 $\mu$ m wide slits or fiber (no slit)
AgPlus Mirrors	Provide reflectivity >95% over the visible and NIR wavelength range and over a wide range of angles of incidence

**Jaz Data Sheet**

Specification	Value
Signal-to-noise ratio	250:1 (full signal)
Integration time	870 $\mu$ s to 65 seconds (20 s typical maximum)
Data transfer rate	~100 scans per second
Optical resolution	~0.3-10.0 nm FWHM
Dark noise	50 RMS counts
Dynamic range	1300:1 for single acquisition
<b>OLED Display Module (JAZ-DPU)</b>	
OLED Display	128 x 64 OLED
Keypad	Pushbutton functions; functionality can be rotated 180°
Onboard computer	Embedded processor with data processing and storage capability
<b>Ethernet &amp; Memory Module (JAZ-E)</b>	
Ethernet	100 Mbps; IEEE 802.3-compliant 10/100
Data Storage	Via SD card
<b>Battery &amp; Memory Module (JAZ-B)</b>	
Type	Rechargeable lithium-ion
Life	8 hours before recharging in JAZ-COMBO configuration. Less when there is more than 1 spectrometer module, or a UV/VIS (~2 hours), VIS/NIR (~6 hours), PX or NEOFOX module present.
<b>Jaz LED Modules</b>	
JAZ-L365	365 nm
JAZ-L405	405 nm
JAZ-L450	470 nm
JAZ-L590	590 nm
JAZ-L640	640 nm
JAZ-LWHITE	White LED
<b>Jaz Industrial Module (JAZ-INDY)</b>	
Communications	RS-232, RS-485
Update Rate	70 Hz maximum (via Ethernet)
Analog Inputs	4 single-ended (+/- 5V) or 2 differential pairs (10V)
Analog Outputs Accuracy Nonlinearity Noise	4 x $\pm$ 5V (software configurable to 0-5V), 16-bit (0.15mV/bit) 2mV for $\pm$ 4.7V, 6mV for $\pm$ 5V <1% full scale typical 6 $\mu$ V
Digital I/O	8; Source 5V, 5 mA; sink 20 mA (TTL compatible input levels)
Current Loop Transmitter	1 x 4-20 mA current loop 2-wire transmit; 8.5 to 36VDC compliance; 14 bit A/D resolution

Specification	Value
Receiver	1-wire receive; 14 bit A/D resolution; over-current protected receiver
Excitation Source	Capable of supplying 10-20V
Connectivity	
RS-232	300-115K Baud, +/- -5V
RS-485	0.5-8M Baud (in multiples of 2)
<b>NeoFox Sport (JAZ-NEOFOX)</b>	
Sampling rate	Acquires data samples every 0.1 second
Battery Life <sup>1</sup>	In the NeoFox Sport configuration, a fully charged, gently used battery will last for approximately three hours.
Average Input Power <sup>1</sup>	
5V (external power)	2.30 W (typical)
3.3 V (Jaz battery)	2.60 W
Operating Temperature	0 – 55 °C
4-20 mA Output Accuracy	160 nA (at 25 °C)
0-5V Output Accuracy	50 m V (at 25 °C)
Jaz Data Logging Rate <sup>1</sup>	1 Hz
Jaz Data Refresh Rate <sup>1</sup>	1 Hz
Standalone Data Refresh Rate	10 Hz
<b>PX Module (JAZ-PX)</b>	
Type	Pulsed Xenon
Bulb life	> 4x10 <sup>8</sup> flashes to 50% of initial intensity
Wavelength Range	190 – 1100 nm
<b>VIS-NIR Light Source Module (JAZ-VIS-NIR)</b>	
Type	Tungsten-halogen
Bulb life:	
At 1025	~25K
At max. intensity (4095)	500 hours
Wavelength Range	400 – 1100 nm
<b>UV-VIS light Source Module (JAZ-UV-VIS)</b>	
Type	Deuterium-tungsten-halogen
Bulb life	~1500 hours
Wavelength Range	Deuterium: 200 – 400 nm; Tungsten: 400 – 1100 nm
<sup>1</sup> Tested in NeoFox Sport configuration (DPU/Battery/NeoFox). Subject to change in other configurations.	

# Mechanical Diagrams

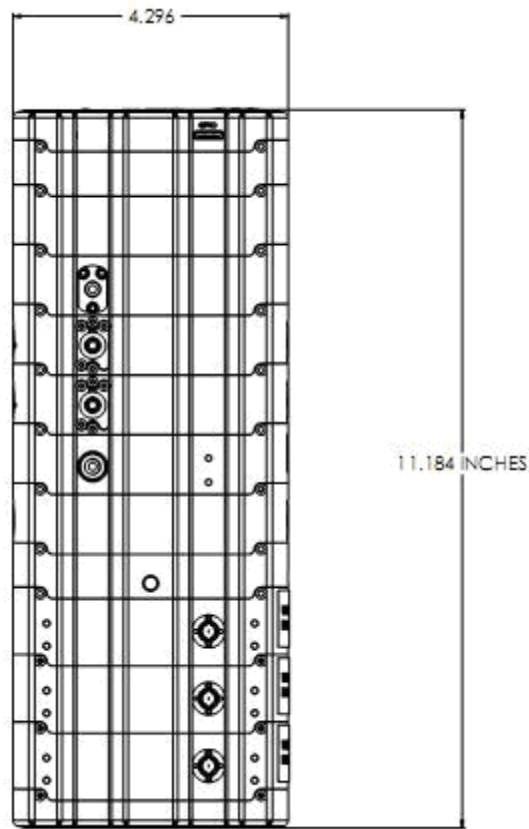


Figure 1: Example Jaz Stack Dimensions (Side View)

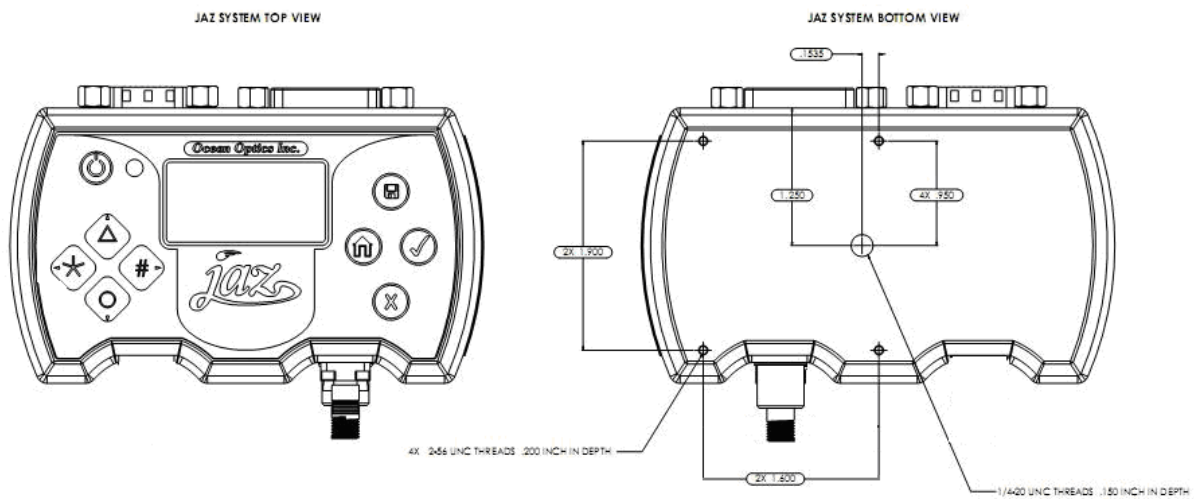


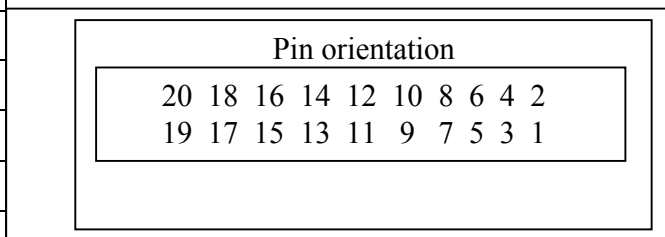
Figure 2: Jaz Top and Bottom View

# Electrical Pinout

## JAZ DPU

Listed below is the pin description for the Accessory Connector located on the Jaz DPU (MIN) 013-20000-100. The connector is model 501014-2071 from Molex.

Pin#	Signal Name	Description
1	FPGASPI_CLK	(3.3v Logic)
2	SPICS_OUT	(3.3v Logic)
3	5V	5 V accessory power
4	TX	RS-232 Data In
5	RX	RS-232 Data Out
6	LAMP_ENABLE	(3.3v Logic)
7	CONT_STROBE	(3.3v Logic)
8	GND	GND
9	EXT-TRIG-IN	(3.3v Logic)
10	SINGLE_STROBE	(3.3v Logic)
11	SCL1	I2C signals
12	SDA1	I2C signals
13	FPGAMOSI	(3.3v Logic)
14	FPGAMISO	(3.3v Logic)
15	GPIO-6	(3.3v Logic)
16	GPIO-5	(3.3v Logic)
17	GPIO-7	(3.3v Logic)
18	GPIO-4	(3.3v Logic)
19	GPIO-2	(3.3v Logic)
20	GPIO-3	(3.3v Logic)



## JAZ DPU GPIO

Listed below is the pin description for the Accessory Connector located on the Jaz DPU GPIO. The connector is model MHDMMR-19-02-F-TH-L-TR from Samtec. Connector information and samples are available from [www. www.Samtec.com](http://www.Samtec.com).

Pin#	Signal Name	Description
1	GND	GND
2	Rx	RS-232 Data In
3	Tx	RS-232 Data Out
4	GND	GND
5	EXT_TRIG_IN	(3.3v Logic)
6	LAMP_ENABLE	GP-OUT
7	GND	GND
8	CONT_STROBE	(3.3v Logic)
9	SINGLE_STROBE	(3.3v Logic)
10	GND	GND
11	SPI_CLK	(3.3v Logic)
12	SPI_CS	(3.3v Logic)
13	MOSI	(3.3v Logic)
14	MISO	(3.3v Logic)
15	GPIO-4	(3.3v Logic)
16	GPIO-3	(3.3v Logic)
17	GPIO-3	(3.3v Logic)
18	5V_GPIO	5V accessory power
19	GPIO-1	(3.3v Logic)

Pin orientation

19 18 17 16 15 14 12 11 10 9 8 7 6 5 4 3 2 1

Function	Input/Output	Description
RS232 Tx	Output	RS232 Transmit signal – for communication with PC connect to DB9 pin 2
RS232 Rx	Input	RS232 Receive signal – for communication with PC connect to DB9 pin 3.
Lamp Enable	Output	A TTL signal that is driven Active HIGH when the Lamp Enable command is sent to the Jaz



Function	Input/Output	Description
Continuous Strobe	Output	TTL output signal used to pulse a strobe that is divided down from the Master Clock signal
Ground	Input/Output	Ground
Single Strobe	Output	TTL output pulse used as a strobe signal, which has a programmable delay relative to the beginning of the spectrometer integration period.
ExtTrigIn	Input	The TTL input trigger signal. In External Hardware Trigger mode this is a rising edge trigger input. In Software Trigger Mode this is an Active HIGH Level signal. In External Synchronization Mode this is a clock input, which defines the integration period of the spectrometer.
GPIO(1-4)	Input/Output	8 2.5V General Purpose Software Programmable Digital Inputs/Outputs
MOSI	Output	The SPI Master Out Slave In (MOSI) signal for communications to other SPI peripherals
MISO	Input	The SPI Master In Slave Out (MISO) signal for communications to other SPI peripherals
SPI CLK	Output	The SPI Clock signal for communications to other SPI peripherals
SPICS_OUT	Output	The SPI Chip/Device Select signal for communications to other SPI peripherals

## Jaz DPU Overview

Jaz is a miniature instrument platform that consists of a Display and Processing unit (DPU) and one or more modules that attach to form a stack. The DPU module provides data connectivity via a miniature USB receptacle, a stack connector that couples with an expansion module and a miniature general purpose connector. A header on the DPU is available to connect to an Ethernet expansion module.

## CCD Overview

### CCD Detector

The detector used for the Jaz is a charge transfer device (CCD) that has a fixed well depth (capacitor) associated with each photodetector (pixel).

Charge transfer, reset and readout initiation begin with the integration time clock going HIGH. At this point, the remaining charge in the detector wells is transferred to a shift register for serial transfer. This process is how the array is read.

The reset function recharges the photodetector wells to their full potential and allows for nearly continuous integration of the light energy during the integration time, while the data is read out through serial shift registers. At the end of an integration period, the process is repeated.

When a well is fully depleted by leakage through the back-biased photodetector, the detector is considered saturated and provides the maximum output level. The CCD is a depletion device and thus the output signal is inversely proportional to the input photons. The electronics in the Jaz invert and amplify this electrical signal.

## CCD Well Depth

We strive for a large signal-to-noise (S:N) in optical measurements so that small signal variations can be observed and a large dynamic range is available. The S:N in photon noise-limited systems is defined and measured as the square root of the number of photons it takes to fill a well to saturation. In Jaz, the well depth of the CCD pixels is about 160,000 photons, providing a S:N of 400:1 (S:N can also be measured as the saturation voltage divided by near-saturation RMS noise). There is also a fixed readout noise component to all samples. The result is a system with a S:N of ~275:1.

There are two ways to achieve a large S:N (e.g., 6000:1) in CCD detectors where photon noise is predominant.

1. Use a large-well device that integrates to saturation over a long period of time until the photon noise is averaged out by the root of  $n$  multiples of a defined short  $\Delta t$ .
2. Use a small-well device that integrates to saturation at one short  $\Delta t$  and then signal average mathematically  $n$  times.

Theoretically, both approaches achieve the same results, though there are large differences in actual operation. Traditional spectroscopic instruments use large-well devices and 16-bit ADCs to achieve the defined S:N. The Jaz uses a small-well device and utilizes signal averaging to achieve the same S:N. A brief comparison of large and small-well devices is shown in the table below.

**Well Depth Comparison**

Large-well CCDs	Small-well CCDs
Low photon noise	Medium photon noise that can be averaged out
Low optical sensitivity	High optical sensitivity
High power consumption	Low power consumption
>10 MHz operating speeds	Moderate operating speeds (~2 MHz)

## Signal Averaging

Signal averaging is an important tool in the measurement of spectral structures. It increases the S:N and the amplitude resolution of a set of samples. The types of signal averaging available in our software are time-based and spatial-based.

When using the time-base type of signal averaging, the S:N increases by the square root of the number of samples. Signal averaging by summing is used when spectra are fairly stable over the sample period. Thus, a S:N of 2500:1 is readily achieved by averaging 100 spectra.

Spatial averaging or pixel boxcar averaging can be used to improve S:N when observed spectral structures are broad. The traditional boxcar algorithm averages  $n$  pixel values on each side of a given pixel.

Time-based and spatial-based algorithms are not correlated, so therefore the improvement in S:N is the product of the two processes.

In review, large-well devices are far less sensitive than small-well devices and thus, require a longer integration time for the same output. Large-well devices achieve a good S:N because they integrate out photon noise. Small-well devices must use mathematical signal averaging to achieve the same results as large-well devices, but small-well devices can achieve the results in the *same period of time*. This kind of signal averaging was not possible in the past because analog-to-digital converters and computers were too slow.

Large-well devices consume large amounts of power, resulting in the need to build thermoelectric coolers to control temperature and reduce electronic noise. Then, even more power is required for the temperature stabilization hardware. But small-well devices only need to use signal averaging to achieve the same results as large-well devices, and have the advantages of remaining cool and less noisy.

## Internal Operation

### Pixel Definition

A series of pixels in the beginning of the scan have been covered with an opaque material to provide an estimate of the baseline signal as it varies due to thermally induced drift. As a Jaz spectrometer module warms up, the baseline signal will shift slowly downward a few counts depending on the external environment. The baseline signal is set around 1500 counts at the time of manufacture. If the baseline signal is manually adjusted, it should be left high enough to allow for system drift. The following is a description of all of the pixels:

**Pixels on the Device**

Pixel	Description
0–11	Not usable
12–29	Optical black pixels
30–31	Not usable
32–2079	Optical active pixels
2080–2085	Not usable

**Pixels Read from the Device via USB**

Pixel	Description
0–17	Optical black pixels
18–19	Not usable
20–2047	Optical active pixels

It is important to note that the Jaz only digitizes the first 2048 pixels.

## CCD Detector Reset Operation

At the start of each integration period, the detector transfers the signal from each pixel to the readout registers and resets the pixels. The total amount of time required to perform this operation is  $\sim 3 \mu\text{s}$ . The user needs to account for this time delay when the pixels are optically inactive, especially in the external triggering modes.

## Digital Inputs & Outputs

### Logic Signals

Provide logic signal input or output as configured by the internal software.

Signal levels are:

Low: 0 - 1.0 v

High: 2.5 - 3.3 v

### 5-Volt Accessory Power

Provides 4.5 to 5.5 volts up to 50 mA for user-defined application. This output is supplied through a diode to protect the internal circuitry.

## Communication and Interface

### USB 2.0

480-Mbit Universal Serial Bus allows for ultra fast data transfer. This is the main communication standard for PC users. The USB bus also provides power as well as communications over a single cord without any bulky external power supplies. USB is limited to 2.5W per device.

### RS-232

The RS232 pins are interfaced with a EIA562 chip, which is like RS-232, but lower voltage ( $\pm 3.7$  volts vs.  $\pm 5$  volts). (These may change to  $\pm 5$  volt output depending on interface chip selection.)

Input Low:  $< 0.6 \text{ v}$

Input High:  $> 2.4 \text{ v}$

Output Low:  $< -3.7 \text{ v}$

Output High:  $> 3.7 \text{ v}$

## I2C

SCL1 and SA1 are logic signals, as described in [Logic Signals](#), that are used for I2C communications. These lines are connected directly to the Blackfin processor.

## SPI

Serial Peripheral Interface is also a widely used communication standard in embedded systems applications.

# Jaz USB Port Interface Communications and Control Information

## Overview

The Jaz is a microcontroller-based Miniature Fiber Optic Spectrometer that can communicate via the Universal Serial Bus, Ethernet (if an Ethernet module is used), or RS-232. This section contains the necessary command information for controlling the Jaz via the USB interface. This information is only pertinent to users who wish to not utilize Ocean Optics' 32-bit (or 64-bit) driver to interface to the Jaz. Only experienced USB programmers should attempt to interface to the Jaz via these methods.

## Hardware Description

The Jaz utilizes a Cypress CY7C68013 microcontroller that has a high speed 8051 combined with an USB2.0 ASIC. Program code and data coefficients are stored in external E<sup>2</sup>PROM that are loaded at boot-up via the I<sup>2</sup>C bus. The microcontroller has 16K of internal SRAM and 64K of external SRAM. Maximum throughput for spectral data is achieved when data flows directly from the external FIFOs directly across the USB bus. In this mode the 8051 does not have access to the data and thus no manipulation of the data is possible.

## USB Information

Ocean Optics Vendor ID number is 0x2457 and the Product ID is 0x2000.

## Instruction Set

### 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 Jaz through End Point 1 Out (EP1). All spectra data is acquired through End Point 2. In and all other queries are retrieved through End Point 1 In (EP1). The endpoints enabled and their order is:

Pipe #	Description	Type	Hi Speed Size (Bytes)	Full Speed Size (Bytes)	Endpoint Address
0	End Point 1 Out	Bulk	512	64	0x01
1	End Point 2 In	Bulk	512	64	0x82
2	End Point 6 In	Bulk	512	64	0x86
3	End Point 1 In	Bulk	512	64	0x81

## USB Command Summary

The following command set is recognized over both the USB and Ethernet ports.

Command Byte Value	Command Name	Command Data	Return Data	Notes
0x01	Initialize	N/A	N/A	Accepted and ignored by Jaz
0x02	Set Integration Time	4 bytes for time in microseconds ( $\mu$ sec), LSB first up to MSB	N/A	Not forwarded to <i>spectrometer</i> until 0x09 (spectrum request) is received
0x03	Set Strobe Enable Status	2 bytes (LSB MSB): LSB 0 for off, 1 for on	N/A	Not forwarded to <i>spectrometer</i> until 0x09 (spectrum request) is received. <i>spectrometer</i> toggles the lamp when appropriate; this may affect any lamp in the Jaz stack. Each lamp can be assigned to a spectrometer, so this causes assigned lamps to change.
0x04	Set Shutdown Mode	2 bytes (LSB MSB)	N/A	Accepted and ignored by Jaz
0x05	Get info	1 byte for info slot number	17 bytes	Return data is 0x05 followed by slot number and 15 bytes of data. Slot number contents are defined in <a href="#">Spectrometer Module Info Slots</a> .
0x06	Set info	1 byte for info slot number, 15 bytes data to write	N/A	Slot numbers are defined in <a href="#">Spectrometer Module Info Slots</a>
0x09	Request Spectrum	N/A	Spectrum	Pixels are formatted in two successive bytes, LSB and MSB; 2048 total pixels resulting in 4096 bytes to read back.
0xC0	Get number of spectrometers	N/A	1 byte	Queries the number of spectrometer modules in this Jaz stack. If this is zero, using the other legacy commands may result in undefined behavior.

Command Byte Value	Command Name	Command Data	Return Data	Notes
0xC1	Set Current Channel	1 byte for module index	N/A	Any index < the value reported by 0xC0 and $\geq 0$ is valid. All subsequent commands will refer to the selected channel. The default is channel index 0 if it exists.
0xC6	Get Jaz Info	1 byte for slot number	17 bytes	Used for retrieving configuration from the Jaz mainboard instead of a spectrometer module. See <a href="#">See Jaz DPU Module Info Slots</a> .
0xC7	Set Jaz Info	1 byte for slot number 15 bytes of data	N/A	Used for storing configuration on the Jaz mainboard instead of a spectrometer module. See <a href="#">See Jaz DPU Module Info Slots</a> .

### Spectrometer Module Info Slots


Index	Name	Description
0x00	Serial number	Serial number for the Jaz spectrometer module
0x01	Wavecal intercept	Intercept (zero-order term) for the wavelength calibration polynomial
0x02	Wavecal 1 <sup>st</sup> order	1 <sup>st</sup> -order (slope) term for the wavelength calibration polynomial
0x03	Wavecal 2 <sup>nd</sup> order	2 <sup>nd</sup> -order term for the wavelength calibration polynomial
0x04	Wavecal 3 <sup>rd</sup> order	3 <sup>rd</sup> -order term for the wavelength calibration polynomial
0x05	Stray light coefficient	Stores stray light constant and possibly slope term (null-separated)
0x06	Nonlinearity 0 order	Intercept (zero-order term) for the nonlinearity calibration polynomial
0x07	Nonlinearity 1 <sup>st</sup> order	1 <sup>st</sup> -order term for the nonlinearity calibration polynomial
0x08	Nonlinearity 2 <sup>nd</sup> order	2 <sup>nd</sup> -order term for the nonlinearity calibration polynomial
0x09	Nonlinearity 3 <sup>rd</sup> order	3 <sup>rd</sup> -order term for the nonlinearity calibration polynomial
0x0A	Nonlinearity 4 <sup>th</sup> order	4 <sup>th</sup> -order term for the nonlinearity calibration polynomial
0x0B	Nonlinearity 5 <sup>th</sup> order	5 <sup>th</sup> -order term for the nonlinearity calibration polynomial
0x0C	Nonlinearity 6 <sup>th</sup> order	6 <sup>th</sup> -order term for the nonlinearity calibration polynomial
0x0D	Nonlinearity 7 <sup>th</sup> order	7 <sup>th</sup> -order term for the nonlinearity calibration polynomial

Index	Name	Description
0x0E	Nonlinearity order	Order of the nonlinearity calibration polynomial, either 4 or 7. This defines how many of the preceding slots are used.
0x0F	Bench	Description of the spectrometer bench. Stores grating number, filter wavelength, and slit size in microns. Fields are space-separated.
0x10	Configuration	Extended description of the spectrometer optics, including detector coating manufacturer, whether a lens is attached, and array wavelength.
0x11	Auto-nulling settings	<p>Baseline and saturation values for auto-nulling. Total of 6 bytes of data:</p> <ol style="list-style-type: none"> <li>1. Autonulling enabled (do not set)</li> <li>2. Temperature compensation enabled (do not set)</li> <li>3. Dark level LSB</li> <li>4. Dark level MSB</li> <li>5. Saturation level LSB</li> <li>6. Saturation level MSB</li> </ol> <p>The first two bytes should not be used until further development is done in the FPGA. The dark level and saturation level corrections work without setting the first 2 bytes.</p>

### Jaz DPU Module Info Slots

Index	Name	Description	Additional Data (Writing) / Return Data (Reading)
0x00	Write-enable	Proprietary	
0x01	Jaz main board serial number	Read-only unless write-enable has been performed.	15 ASCII bytes
0x10	MAC address	6 bytes representing 48-bit address. Read-only unless write-enable has been performed.	6 bytes representing 48-bit address
0x11	IPv4 address	4 bytes representing 32-bit IPv4 address. If all 4 bytes are 0x00 then DHCP is to be used for the IPv4 address, netmask, gateway, and nameserver. After this block of 4 bytes is another block of 4 bytes containing the current IP address (or 0x00 if none is defined). Allows the active IP for a Jaz using DHCP to be queried.	4 bytes representing 32-bit address
0x12	IPv4 netmask	4 bytes representing 32-bit IPv4 netmask, only defined if not using DHCP. After this block of 4 bytes is another block of 4 bytes containing the current IP address (or 0x00 if none defined). This allows the active netmask for a Jaz using DHCP to be queried.	4 bytes representing 32-bit address



Index	Name	Description	Index
0x13	IPv4 gateway	4 bytes representing 32-bit IPv4 gateway (router) address, only defined if not using DHCP (and not strictly necessary if Jaz is restricted to local network).	4 bytes representing 32-bit address
0x14	IPv4 nameserver	4 bytes representing the IPv4 address of a computer running a DNS service. To be effective, the nameserver must be accessible with the current netmask and gateway. Only defined if not using DHCP.	4 bytes representing 32-bit address
0x20	Time and date	4 bytes, MSB first, representing the number of seconds elapsed since 1-Jan-1970. This must agree with the <i>time()</i> function in <i>time.h</i> in standard C. Read and write are supported (no write-enable required).	4 Bytes: MSWMSB   MSWLSB   LSWMSB   LSWLSB of seconds since epoch
0x21	Keypad type	<p>1 byte for the keypad type:</p> <p>0: "Normal" keypad. The OLED is at the top of the keypad and the USB and DC power connectors point away from the user</p>  <p>1: "Away" keypad. The OLED is at the bottom of the keypad and the USB and DC power connectors point toward the user.</p> <p>Read-only unless write-enable has been performed. Note that Jaz will store the keypad type in flash and continue to use it until commanded otherwise. Factory-set.</p>	1 Byte: 0 for normal keypad and 1 for Away keypad
0x24	Default RS232 Rate	1 Byte. Acceptable Values = 0-6	
0x60	Lamp Enable	2 Bytes: MSB   LSB of Lamp Enable (LSB is 0 or 1). Acceptable Values = 0-1	
0x61	Trigger Source	<p>4 Bytes: MSB   LSB of Trigger Source Acceptable Values = 0-4</p> <p>0: External Trigger pin (synchronized to DPU FPGA clock)</p> <p>1: Reserved</p> <p>2: Logic 0</p> <p>3: Logic 1</p> <p>4: Internal Trigger Generator (see Slot 0x62 below)</p>	
0x62	Internal Trigger Period	4 Bytes: MSWMSB   MSWLSB   LSWMSB   LSWLSB of Trigger Period. Acceptable Values = 0x000C – 0xFFFF	

Index	Name	Description	Index
0x64	Single Strobe High Transition	2 Bytes: MSB   LSB of Single Strobe High Transition Value. Acceptable Values = 0x000C – 0xFFFF	
0x65	Single Strobe Low Transition	2 Bytes: MSB   LSB of Single Strobe Low Transition Value. Acceptable Values = 0x000C – 0xFFFF	
0x66	Continuous Strobe period	4 Bytes: MSWMSB   MSWLSB   LSWMSB   LSWLSB of Strobe Period. Acceptable Values = 0x000C – 0xFFFF	
0x68	GPIO Mux (Direction) Bit Set	2 Bytes: MSB   LSB of GPIO Mux Bitset. Acceptable Values = 0x0000 – 0x003C	
0x69	GPIO Output Enable Bit Set	2 Bytes: MSB   LSB of GPIO Output Enable Bitset. Acceptable Values = 0x0000 – 0x003C	
0x6A	GPIO Data Bit Set	2 Bytes: MSB   LSB of GPIO Data Register. Acceptable Values = 0x0000 – 0x003C	
0xA3	Primary uClinux time stamp	Format is 4 bytes, MSB first, for the number of seconds elapsed since 1-Jan-1970 (i.e., UNIX time). Read-only.	1 Byte
0xB0	Main board FPGA version	2 bytes corresponding to 16-bit firmware version register in FPGA. Read-only.	2 Bytes: MSB   LSB of Lamp Enable (LSB is 0 or 1)
0xC0	Module count	Number of modules detected below the DPU (1 byte, 0x00-0x0F). Read-only.	4 Bytes: MSB   LSB of Trigger Source
0xCN	Module N info	For the Nth module (from 1 to F, inclusive) this reports the following information (read-only): Module type (byte) 0xB0: Spectrometer 0x21: Ethernet/SD 0x31: Lamp or LED 0x41: Oxygen sensor 0x51: Battery/SD Module variant (byte) Stack interface CPLD/FPGA firmware version (2 bytes) Other configuration data (up to 11 more bytes, defined on a per-module basis)	4 Bytes: MSWMSB   MSWLSB   LSWMSB   LSWLSB of Trigger Period

## Set Integration Time

Sets the Jaz integration time in microseconds. The value is a 32-bit value whose acceptable range is 1000 – 65,535,000 $\mu$ s. If the value is outside this range the value is unchanged. The integration clock allows 4 significant digits to be specified, with a maximum resolution of 10 $\mu$ s. For example,

1 000 – 99 990  $\mu$ s at 10 $\mu$ s  
 100 000 – 999 990  $\mu$ s at 100  $\mu$ s  
 1 000 000 – 9 999 000  $\mu$ s at 1 000  $\mu$ s  
 10 000 000 – 65 530 000  $\mu$ s at 10 000  $\mu$ s

### Byte Format

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4
0x02	LSW-LSB	LSW-MSB	MSW-LSB	MSW-LSB

MSW & LSW: Most/Least Significant Word

MSB & LSB: Most/Least Significant Byte

## Set Strobe Enable Status

Sets the Jaz Lamp Enable line (J2 pin 4) as follows. The Single Strobe and Continuous Strobe signals are enabled/disabled by this Lamp Enable Signal. It may also enable lamps in the stack that are assigned to the last selected spectrometer module (external pins must also be assigned to a spectrometer module.)

Data Byte = 0 → Lamp Enable Low/Off  
 Data Byte = 1 → Lamp Enable HIGH/On

### Byte Format

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

## Set Shutdown Mode

Ignored.

### Byte Format

Byte 0	Byte 1	Byte 2
0x04	Data byte LSB	Data Byte MSB

## Get Info

Queries any of the 20 stored spectrometer configuration variables. The Query command is sent to End Point 1 Out and the data is retrieved through End Point 1 In. When using Query Information to read EEPROM slots, data is returned as ASCII text. However, everything after the first byte that is equal to numerical zero will be returned as garbage and should be ignored.

The 20 configuration variables are indexed as follows:

**Data Byte - Description**

- 0 – Serial Number
- 1 – 0<sup>th</sup> order Wavelength Calibration Coefficient
- 2 – 1<sup>st</sup> order Wavelength Calibration Coefficient
- 3 – 2<sup>nd</sup> order Wavelength Calibration Coefficient
- 4 – 3<sup>rd</sup> order Wavelength Calibration Coefficient
- 5 – Stray light constant
- 6 – 0<sup>th</sup> order non-linearity correction coefficient
- 7 – 1<sup>st</sup> order non-linearity correction coefficient
- 8 – 2<sup>nd</sup> order non-linearity correction coefficient
- 9 – 3<sup>rd</sup> order non-linearity correction coefficient
- 10 – 4<sup>th</sup> order non-linearity correction coefficient
- 11 – 5<sup>th</sup> order non-linearity correction coefficient
- 12 – 6<sup>th</sup> order non-linearity correction coefficient
- 13 – 7<sup>th</sup> order non-linearity correction coefficient
- 14 – Polynomial order of non-linearity calibration
- 15 – Optical bench configuration: gg fff sss  
gg – Grating #, fff – filter wavelength, sss – slit size
- 16 – Jaz configuration: AWL V  
A – Array coating Mfg, W – Array wavelength (VIS, UV, OFLV), L – L2 lens installed, V – CPLD Version
- 17 – Autonulling
- 18 – Reserved
- 19 – Reserved

### Byte Format

Byte 0	Byte 1
0x05	Data byte

### Return Format (EP1)

The data is returned in ASCII format and read in by the host through End Point 1.

Byte 0	Byte 1	Byte 2	Byte 3	...	Byte 17
0x05	Configuration Index	ASCII byte 0	ASCII byte 1	...	ASCII byte 15

## Set Info

Writes any of the 20 stored spectrometer configuration variables to EEPROM. The 20 configuration variables are indexed as described for Get Info (above). The information to be written is transferred as ASCII characters.

### Byte Format

Byte 0	Byte 1	Byte 2	Byte 3	...	Byte 17
0x06	Configuration Index	ASCII byte 0	ASCII byte 1	...	ASCII byte 15

## Request Spectrum

Initiates a spectrum acquisition. Jaz will acquire a complete spectrum (2048 pixel values). The data is returned in bulk transfer mode through EP2. The pixel values are decoded as described below.

### Byte Format

Byte 0
0x09

### Return Format

The format for the returned spectral data is dependant upon the USB communication speed. The format for both High Speed (480 Mbps) and Full Speed (12Mbps) is shown below. All pixel values are 16 bit values which are organized in LSB | MSB order.

#### USB High Speed (480Mbps) Packet Format

The data is read from EP2In. The packet format is described below.

Packet #	End Point	# Bytes	Pixels
0	EP2In	512	0-255
1	EP2In	512	256-511
2	EP2In	512	512-767
3	EP2In	512	768-1023
4	EP2In	512	1024-1279
5	EP2In	512	1280-1535
...	EP2In	512	
8	EP2In	512	1792-2048

The format for the first packet is as follows (all other packets have a similar format except the pixel numbers are incremented by 256 pixels for each packet).

#### Packet 0

Byte 0	Byte 1	Byte 2	Byte 3
Pixel 0 LSB	Pixel 0 MSB	Pixel 1 LSB	Pixel 1 MSB

...		Byte 510	Byte 511
		Pixel 255 LSB	Pixel 255 MSB

### **USB Full Speed (12Mbps) Packet Format**

In this mode all data is read from EP2In. The pixel and packet format is shown below.

Packet #	End Point	# Bytes	Pixels
0	EP2In	64	0-31
1	EP2In	64	32-63
2	EP2In	64	64-95
...	EP2In	64	
63	EP2In	64	2016-2047

#### **Packet 0**

Byte 0	Byte 1	Byte 2	Byte 3
Pixel 0 LSB	Pixel 0 MSB	Pixel 1 LSB	Pixel 2 MSB
...		Byte 62	Byte 63
		Pixel 31 LSB	Pixel 31 MSB

### **Autonulling**

Slot 0x11 (17) contains autonulling information that has a scaling term used to adjust the magnitude of the entire spectrum. This can be read out by sending bytes 0x05 11 to the low-speed out endpoint (0x01) and then reading out 17 bytes from the low-speed in endpoint (0x81). The bytes of use are Bytes 7 and 8. The 17 bytes will be formatted as follows:

0x05 11 XX XX XX XX SS SS XX XX XX XX XX XX XX XX

Where:

XX = reserved bytes (see [Spectrometer Module Info Slots](#) for more information)

SS = saturation level of the device as two bytes (LSB followed by MSB).

These need to be assembled into a single 16-bit value. Any time that a spectrum is read from the spectrometer, each pixel's intensity value should be multiplied by (65535.0/saturation\_level) to set the scale appropriately.

The contents of slot 0x11 are set at the factory and should not be altered.

### **Get Number of Spectrometers**

Queries the number of spectrometer modules in the Jaz stack. If 0 is returned, using the other commands can result in undefined behavior.

## Byte Format

Byte 0
0xC0

## Set Current Channel

Any index less than the value reported by 0xC0 and greater than or equal to zero is valid. All subsequent commands will refer to the selected channel. The default is channel index 0 if it exists. Byte 1 indicates which spectrometer channel should be selected.

### Byte Format

Byte 0	Byte 1
0xC1	0x00 – 0x0F

## Get Jaz Info

Retrieves configuration from the Jaz mainboard instead of a spectrometer module. See [Jaz DPU Module Info Slots](#).

This command is sent to End Point 1 Out and the data is retrieved through End Point 1 In.

### Byte Format

The data is sent through End Point 1 Out.

Byte 0	Byte 1
0xC6	Register

### Return Format (EP1)

The data is returned through End Point 1 In.

Byte 0	Byte 1	Byte 2	Byte 3	...	Byte 16
0xC6	Register	Data byte 0	Data byte 1	...	Data byte 15

## Set Jaz Info

Writes data to the Jaz mainboard configuration variables that are not read-only. See [Jaz DPU Module Info Slots](#).

This command is sent to End Point 1.

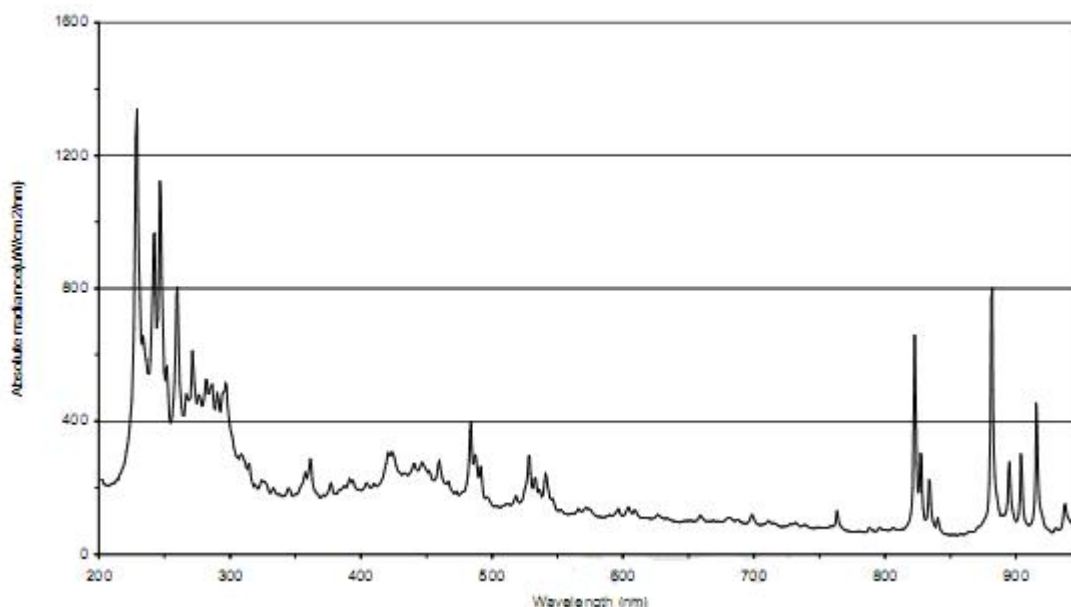
### Byte Format

Byte 0	Byte 1	Byte 2	...	Byte 16
0xC7	Register	Data Byte 0	...	Data byte 15

# JAZ-PX Module

## Overview

The JAZ-PX lamp is a pulsed, short arc xenon lamp for UV-VIS applications such as absorbance, bioreflectance, fluorescence and phosphorescence. The lamp has a specified pulse frequency of 200 Hz (maximum 500 Hz) and spectral response from 190 – 1100 nm.



**Absolute Irradiance Spectrum from Typical JAZ-PX Lamp Flash**

## Timing Signals

When operating in a Jaz configuration, the JAZ-PX module has a number of useful features. Most significantly, the device includes a configurable timing generator, which controls flash timing with high precision and limited maintenance from the user. Rather than repeatedly providing an explicit trigger to flash the lamp, the user specifies behavior attributes and timing parameters via the SPI interface and the JAZ-PX independently controls the lamp's trigger. The JAZ-PX also integrates an intensity control, non-volatile memory, and an in-field [Battery Monitor](#) feature.

## Intensity Control

The PX\_INTENSITY register controls the output voltage of the flash lamp's output capacitor. This can be used to vary the intensity of the lamp's optical output. The voltage range is 400V to 600V. The onboard DAC that controls this output voltage has 256 bit resolution and the output tolerance is a fairly lenient  $\pm 15V$ . The valid input values for this register are from 0x0000 → 0xFFFF, although due to the limited resolution of the DAC, only the high order byte of this register is actually used.



## Non-Volatile Memory

The JAZ-PX module includes  $2^{13}$  16-bit words of non-volatile memory. The first 512 locations are reserved for use by Jaz. Memory access is controlled via the memory access SPI registers as follows:

- **Timing** – Memory writes take less than 60 microseconds and memory reads take less than 50 microseconds. During this time, bit 0 of the PX\_MEM\_BUSY register will be high. When the operation has completed, the PX\_MEM\_BUSY bit will go low. Operations are triggered by a low to high transition on bit 0 of the PX\_MEM\_TRIGGER register. However, triggers which occur while the PX\_MEM\_BUSY bit is high will be ignored. The PX\_MEM\_TRIGGER bit must be held low for at least 1 microsecond after the PX\_MEM\_BUSY bit goes low and must remain high for at least 1 microsecond in order to ensure a valid operation.  
During an operation, the PX\_MEM\_ADDRESS and PX\_MEM\_DATA\_IN registers should maintain their values for the duration of the operation. Wait until the PX\_MEM\_BUSY bit goes low, signaling the end of the operation, before changing these values.
- **Writing to memory** – Before updating any of the memory control registers, wait for the PX\_MEM\_BUSY bit to go low. To write to a memory address, begin by writing the target address to the PX\_MEM\_ADDRESS register and the new 16 bit value to the PX\_MEM\_DATA\_IN register. Then, write a “1” to bit 0 of the PX\_MEM\_READ\_WRITE register to put the memory controller into write mode. Finally, set bit 0 of the PX\_MEM\_TRIGGER to 0 for at least 1 microsecond, and then set it to 1 to begin writing the value in the PX\_MEM\_DATA\_IN register to the PX\_MEM\_ADDRESS register.
- **Reading from memory** – Before updating any of the memory control registers, wait for the PX\_MEM\_BUSY bit to go low. To read from a memory address, begin by writing the source address to the PX\_MEM\_ADDRESS register. Then, write a “0” to bit 0 of the PX\_MEM\_READ\_WRITE register in to put the memory controller into read mode. Finally, set bit 0 of the PX\_MEM\_TRIGGER to 0 for at least 1 microsecond, and then set it to 1 to begin writing the value in the PX\_MEM\_DATA\_IN register to the PX\_MEM\_ADDRESS register. When the PX\_MEM\_BUSY bit goes low, the data in the PX\_MEM\_DATA\_OUT register will be the output of the read operation.
- **Reserved memory addresses** – Certain memory addresses are used to store manufacturer data. Memory addresses lower than 0x0100 are not writable.

## Battery Monitor

The JAZ-PX has a voltage monitor to ensure that its input voltage does not drop beneath 2.7V for more than 1 ms. When an under-voltage condition occurs, the JAZ-PX will immediately stop flashing and set bit 0 of the PX\_BATTERY\_BAD register to 1. Once this bit is set, the device will not flash again until it has been reset by powering down the device.

## CCA Revision

This register reports the CCA revision code of the JAZ-PX module.

## Serial Number

The serial number is stored in onboard memory in address 2 of the EEPROM as a decimal number. The first unit was serialized as 10001 (decimal) and the rest of the units count up from there.

## Firmware Update

Firmware updates are performed by sequentially writing every byte of a valid ACE file to the Firmware Update Controller. To do this, sequentially send the ACE file byte by byte to the JAZ-PX module via its SPI interface.

Before writing to any of the firmware update registers, wait for the PX\_FW\_BUSY bit to go low. Once the controller is no longer busy, you can write a data byte from the ACE file to the lower order byte of the PX\_FW\_DATA\_IN register. Due to the sensitivity of this process, it's a good idea to read back and verify the value of the register that has just been written. Next, write a 0 to bit 0 of the PX\_FW\_LOAD\_TRIGGER and hold this value for at least one microsecond before writing a 1 to bit 0 of this register. This will cause the PX\_FW\_BUSY bit to go high until the operation has completed. It may take up to 1 microsecond after the transition of the load trigger before the PX\_FW\_BUSY bit goes high. Once the PX\_FW\_BUSY returns to 0, the byte has been processed and this procedure can be repeated for the next byte in the file.

The PX\_FW\_STATUS register has two important flag bits that provide information about the status of the firmware update. The values of this register are as follows:

Register Value	Meaning
0	Firmware update has not yet completed
1	Firmware update has completed without errors
2	Firmware update encountered errors and aborted

### Note

Be extremely careful during the firmware update procedure to ensure that there are no errors during the transfer of information from the ACE file to the PROM device. Errors during this process could result in the permanent failure of the device. If this occurs, the configuration PROM will need to be reflashed via JTAG, which will, in most cases, require it to be returned to Ocean Optics at some cost to the user.

## Timing Generator

To make the JAZ-PX both easy to use and extremely flexible, the module includes a configurable internal timing generator capable of generating the high precision lamp trigger internally. This timing generator reduces the dependence of the JAZ-PX module on external logic, yet also allows the module to be synchronized with other elements of the system.

## Parameters

The three timing parameters for the timing generator (holdoff time, high time, and low time) are set via SPI registers. These parameters are 32 bits each, and their values are given in terms of microseconds. Because the SPI registers are only 16 bits wide, each parameter has a high word register and a low word register, denoted by the HW and LW suffixes. There are also two other registers - a command register and a number of flashes register. The command register affects the behavior of the timing generator as described in [JAZ-PX Command Register](#). The Number of Flashes register affects behavior when in [Triggered Mode](#), as described below.

## Free Running Mode

The pulse timing generator has two basic operating modes: Free Running mode and Triggered mode. This mode is selected by the PX\_BIT\_CMD\_FREE\_RUNNING bit. When operating in the Free Running mode, the JAZ-PX lamp trigger alternates between its high and low states in accordance with the timing parameters. Triggers (external or forced) do not affect the output so the Holdoff Period” and Number of Flashes parameters are irrelevant. Additionally, when the PX\_BIT\_CMD\_FREE\_RUNNING bit toggles from 0 (Triggered mode) to 1 (Free Running mode), the trigger output will immediately go high and the first high cycle timing will begin at that point.

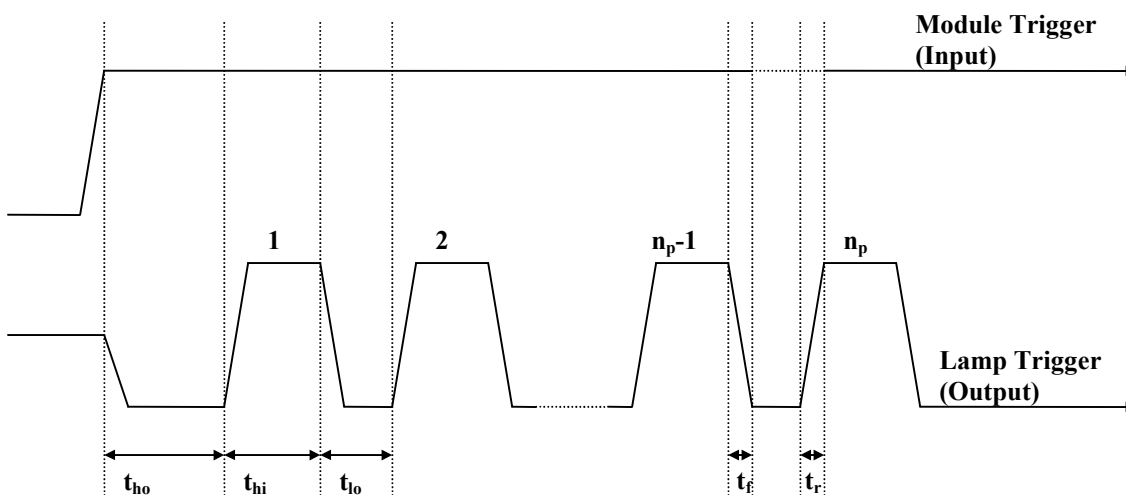
## Triggered Mode

In Triggered mode, a trigger event always causes the device to immediately enter the holdoff state, then alternate between high and low states repeatedly  $n$  times in accordance with the timing parameters (where  $n$  is the value specified by the PX\_WAVE\_NUM\_FLASHES parameter). A trigger event is caused in either of two ways:

- By external trigger, which is generated when the voltage on LVDS\_7P is greater than the voltage on LVDS\_7N in the Jaz stack *and* the PX\_BIT\_CMD\_TRIGGER\_ENABLE bit is set to 1.
- Alternatively, you can force a trigger event by writing a low to high transition on the PX\_BIT\_CMD\_FORCE\_TRIGGER bit. This will have the same effect as an external trigger and will generate a trigger event even if the trigger enable bit is set to 0.

## Triggered Mode – Run Forever

As noted above, when the PX\_BIT\_CMD\_RUN\_FOREVER bit is set to 0, the timing generator issues  $n$  flashes and then stops issuing flashes. However, if this bit is set to 1, the timing generator ignores the PX\_WAVE\_NUM\_FLASHES parameter and continues to issue flashes in accordance with the high and low time parameters indefinitely. This mode can be especially useful to “re-sync” a continuously pulsing light source with some other timer.



### Timing Parameters for the Pulse Timing Generator Module

#### Absolute Maximum Timing Parameters

Symbol	Name	Min	Max	Note/Conditions
$t_{ho}$	Holdoff Time	1 $\mu$ s	4,294,967,295 $\mu$ s	Value 0 is not allowed
$t_{hi}$	High Time	1 $\mu$ s	4,294,967,295 $\mu$ s	Value 0 is not allowed
$t_{lo}$	Low Time	1 $\mu$ s	4,294,967,295 $\mu$ s	Value 0 is not allowed
$t_r, t_f$	Rise/Fall Times	--	21 ns	--
*	Trigger Hold	1 $\mu$ s	--	Not pictured in diagram

#### Recommended Timing Parameters for the JAZ-PX Module

Symbol	Name	Min	Note/Conditions
$t_{hi} + t_{lo}$	Period	2000 $\mu$ s	The 2 ms minimum period guarantees that the output capacitor on the lamp has time to charge fully between flashes. It corresponds to 500 Hz operation.
$t_{hi}$	High Time	10 $\mu$ s	The 10 $\mu$ s high hold time is a requirement of the lamp hardware itself. High and low hold times shorter than 10 $\mu$ s are not recommended.
$t_{lo}$	Low Time	1990 $\mu$ s	The 1990 $\mu$ s low hold time is a requirement of the lamp hardware itself. High and low hold times shorter than 1990 $\mu$ s are not recommended.
$t_{ho}$	Holdoff Time	1990 $\mu$ s	The 1990 $\mu$ s minimum time guarantees that the output capacitor on the lamp has time to charge fully before the first flash after a trigger event. This minimum parameter can be disregarded if the first flash after a trigger will not occur within 1990 $\mu$ s of a previous flash.

## JAZ-PX Register Space and Descriptions

#	Read	Write	Name	Description
0	Y	N	JAZ_IDENTITY	The JAZ-PX identity descriptor always reports 0x3201 (ID 0x32, variant 0x01)
1	Y	Y	PX_COMMAND	See <a href="#">JAZ-PX Command Register</a>
2	Y	Y	PX_WAVE_HOLDOFF_LW	Low order word of $t_{ho}$ value
3	Y	Y	PX_WAVE_HOLDOFF_HW	High order word of $t_{ho}$ value
4	Y	Y	PX_WAVE_HIGH_LW	Low order word of $t_{hi}$ value
5	Y	Y	PX_WAVE_HIGH_HW	High order word of $t_{hi}$ value
6	Y	Y	PX_WAVE_LOW_LW	Low order word of $t_{lo}$ value
7	Y	Y	PX_WAVE_LOW_HW	High order word of $t_{lo}$ value
8	Y	Y	PX_WAVE_NUM_FLASHES	Number of times to flash the lamp after detecting a trigger event ( $n_p$ )
9	Y	Y	PX_INTENSITY	Intensity control. See <a href="#">Intensity Control</a> .
10	Y	Y	PX_MEM_ADDRESS	Bits 13 to 0 store the 14 bit address for memory read or write operations
11	Y	Y	PX_MEM_READ_WRITE	When bit 0 is low, the memory controller will perform read operations when triggered. When bit 1 is high, the memory controller will perform write operations when triggered.
12	Y	Y	PX_MEM_TRIGGER	A low to high transition on bit 0 will cause the memory controller to perform a single read or write operation if it is not already busy. The trigger is ignored if it occurs while bit 0 of the PX_MEM_BUSY register is high.
13	Y	N	PX_MEM_DATA_OUT	Holds the most recent data read from memory
14	Y	Y	PX_MEM_DATA_IN	Holds the data that will be written to memory at the next memory operation trigger
15	Y	N	PX_MEM_BUSY	Bit 0 = 1 indicates that the previous memory operation has not yet completed. Bit 0 = 0 indicates that the memory is ready for the next instruction.
21	Y	N	PX_CCA_REVISION	Reads back the CCA revision code
22	Y	N	PX_BATTERY_BAD	Bit 0 will be set to 1 by the hardware if an undervoltage condition is detected. The lamp will not flash again until power has been cycled.
68	Y	N	JAZ_FIRMWARE	Always reports firmware version

## JAZ-PX Command Register

Bit	Name	Description
0	Reserved	
1	Reserved	
2	PX_BIT_CMD_OUTPUT_ENABLE	1 – Enables lamp output 0 – Disables lamp output by holding the lamp trigger low
3	PX_BIT_CMD_TRIGGER_ENABLE	1 – Enables the modular trigger input 0 – Disables the trigger input. Writing low to high transition on the PX_BIT_CMD_FORCE_TRIGGER bit will still cause a trigger event.
4	PX_BIT_CMD_RUN_FOREVER	1 – Causes the timing generator to continue asserting flashes regardless of the value of the PX_WAVE_NUM_FLASHES register. 0 – After detecting a trigger event, the timing generator will assert a specific number of flashes, as specified in the PX_WAVE_NUM_FLASHES register.
5	PX_BIT_CMD_FORCE_TRIGGER	0 → 1 Writing a low to high transition on this bit causes the same effect as detecting a low to high transition on the external trigger source. Low to high transitions on this bit cause a trigger event even if the PX_BIT_CMD_TRIGGER_ENABLE is 0.
6	PX_BIT_CMD_FREE_RUNNING	1 – In this mode, the timing generator does not wait for a trigger to begin flashing. Instead it begins immediately. In this mode, the PX_WAVE_NUM_FLASHES and PX_WAVE_HOLDOFF parameters will be ignored. 0 – Normal operation.
7	PX_BIT_CMD_TIMING_RESET	1 – When this bit is set to 1, the PX timing generator state machine will be reset to the idle state. This bit does not reset the device registers, it just causes the timing generator to exit its current state and reset its timing counters. For instance, this feature can be used if the timing generator is stuck in a very long period. 0 – Normal operation.
8-15	Reserved	

## JAZ-PX Quasi-SPI Protocol

When controlled via the Jaz interface, the JAZ-PX will respond to the same register-based Jaz Quasi-SPI interface that is used on all other modules. The primary difference between the Jaz Quasi-SPI and SPI proper is that rather than using the chip select bit as a slave module enable, the first byte of any transfer is used as an address which slave modules use to determine whether or not they have been selected. A secondary difference is that there is a device discovery process in which this module level address is assigned by the host controller to each slave module in the stack. With the exception of the address byte, the protocol supports full duplex transfers.

## Command Format

The Quasi-SPI commands come to the JAZ-PX in the following format, and are generated by the host module, typically the DPU. There is no CRC, frame checking, or error reporting involved in this transfer. Transfers begin when the CS line goes low (start condition) and end when the CS line goes high (stop condition). However, the typical SPI start and stop timing and conditions regarding the states of the MOSI and SPICLK lines in relation to the CS line are disregarded.

Device Address (8)	Register Address (7)	R/W (1)	Data MSB (8)	Data LSB (8)
--------------------	----------------------	---------	--------------	--------------

**DEVICE ADDRESS:** The device address is an 8 bit identifier which was set in the device discovery process, as described later in this document.

**REGISTER ADDRESS:** This is the address of the internal register that the host would like to read from or write to.

**R/W:**

0 – Read

1 – Write.

The JAZ-PX has a shared read/write register address space, which means an address which is used for a writable register will not be used for a readable register with a different data source. However, not all registers are both readable and writable. Refer to module-specific register documentation for details.

**DATA MSB, DATA LSB:** The high byte and low byte of the data packet.

## Device Discovery

The device discovery transfer is a special case subset of the typical SPI transfer packet format that is organized as follows. Only the first transfer after a device reset is handled. Subsequent device discovery transfers are ignored.

Start Byte (8)	New Device Address (7)	Don't Care (16)
----------------	------------------------	-----------------

**START BYTE:** The start byte for a device discovery packet is 0xAE.

**NEW DEVICE ADDRESS:** The device address is given as an 8 bit address.

## SPI CS Input and Output

There is one additional note about the device discovery process. The slave modules in the Jaz stack all have 1 CS input and 1 CS output, which is normally a pass-through of the CS input. However, prior to receiving its address from the host, slave modules should intercept the CS input and hold the CS output high, thus preventing downstream modules from handling SPI commands from the host until it has received its address. In this manner, the modules are “discovered” and assigned addresses sequentially.

# JAZ-PX Module Characteristics and Specifications

## Electrical Requirements

Parameter	Minimum	Maximum	Notes/Conditions
Input Voltage – Jaz: Battery Power Mode External Power Mode	2.7V 4.7V	5.5V 5.5V	200 Hz, 400V 200 Hz, 600V
Enable Signal: On Off	2.4V 0.0V	3.6V 2.0V	Operational mode Shutdown mode
Logic Input: Low Level High Level	0.0V 2.3V	0.7V 3.8V	$V_{IL} = 0.7V$ $V_{IH} = 2.3V$
Jaz Stack Trigger Voltage Level	100 mV	600 mV	Differential Input Level
Jaz Stack Trigger Hold Time	1.5 $\mu$ s	--	Minimum hold time for trigger detection

## Electrical Characteristics

Parameter	Minimum	Typical	Maximum	Notes/Conditions
Shutdown Power	--	--	100 $\mu$ W	Power in shutdown mode
Average Power	--	--	4.5 W	@ $V_{in} = 2.7V$ , 200 Hz, 400V
Peak Power	--	--	5.5 W	@ $V_{in} = 2.7V$ , 200 Hz, 400V
Nonflashing Power	--	--	800 mW	Power consumption with Output Enable off $V_{in} = 5V$ , 400V
Discharge Voltage	400V	400V	600V	$\pm 20V$ accuracy
Jaz Battery Life	--	3 hr	--	Test stack configuration: DPU, battery, JAZ-PX, spectrometer. 200 Hz, 400V Operating the JAZ-PX on battery power at frequencies above 200 Hz or intensities above 400V can result in unexpected behavior from the JAZ-PX module and may affect other modules in the stack.



## Optical Characteristics

Parameter	Minimum	Typical	Maximum	Notes/Conditions
Flash Rate	--	--	200 Hz	Stability and intensity may decrease above 200 Hz
Lamp Life	10 <sup>8</sup> flash	5X10 <sup>9</sup>	--	Number of flashes to 50% of original intensity. Module operation not characterized beyond lamp life.
Spectral Range	190 nm	--	1100 nm	See figure on next page for typical lamp spectrum
Optical Stability	--	1% CV*	--	After 20 consecutive flashes at constant frequency. 200 Hz, 400V
Stability Warm-up Time	--	15	--	Number of flashes before stability
Optical Connector	SMA 905; no filter, lens or slit			--

\*CV (Coefficient of Variation) is defined as the standard deviation of 20 samples divided by the mean of 20 samples.

## Operating Conditions

Parameter	Minimum	Maximum	Notes/Conditions
Operating Temperature	0°C	55°C	Intensity may drift over operating temperature
Operating Humidity	--	95%	Noncondensing
Storage Temperature	-10°C	60°C	--
Shock and Vibration	--	20G	Mounted in Jaz stack assembly

# JAZ-INDY Module

## Overview

The JAZ-INDY is a module for industrial applications offering RS-232 and RS-485 communications. The JAZ-INDY module provides both a two wire 4-20mA current loop transmitter, and a power source and return for implementation of a two wire 4-20mA current loop receiver.

The module has 4 analog I/Os software-configurable to -5 to +5 volts and 8 digital I/Os. Data for analog inputs and outputs can be processed in values such as volts or counts. The module complies with RoHS, CE and FCC Part 15, Class A requirements.

# JAZ-INDY Electrical Pinouts

## J11 Connector

The JAZ-INDY's J11 26-pin connector is used for the 4-20mA current loop, as well as Digital and Analog inputs/outputs.

Pin#	Signal Name	Description
1	4-20 Tx V+	4-20 Analog Loop Transmitter
2	An GND	Analog Ground
3	An In 1	Analog Input 1
4	An Out 4	Analog Output 4
5	An Out 1	Analog Output 1
6	Dig GND	Digital Ground
7	Dig I/O 7	Digital Input/Output 7
8	Dig I/O 4	Digital Input/Output 4
9	Dig I/O 1	Digital Input/Output 1
10	4-20 Tx V-	4-20 Analog Loop Transmitter
11	4-20 Sense	4-20 Current Loop Receiver
12	An GND	Analog Ground
13	An In 2	Analog Input 2
14	An Out 2	Analog Output 2
15	An GND	Analog Ground
16	Dig I/O 8	Digital Input/Output 8
17	Dig I/O 5	Digital Input/Output 5
18	Dig I/O 2	Digital Input/Output 2
19	10 or 20 Vdd	Excitation Source Voltage
20	An GND	Analog Ground
21	An In 4	Analog Input 4
22	An In 3	Analog Input 3
23	An Out 3	Analog Output 3
24	Dig GND	Digital Ground
25	Dig I/O 6	Digital Input/Output 6
26	Dig I/O 3	Digital Input/Output 3

Pin orientation

9	8	7	6	5	4	3	2	1
18	17	16	15	14	13	12	11	10
26	25	24	23	22	21	20	19	

## J12 Connector

The J12 connector is for RS-232 and RS-485 communications. The cable used determines the communication type.

Pin#	Signal Name	Description	Type
1	Z	TX-	RS-485
2	RxD		RS-232
3	TxD		RS-232
4	A	RX+	RS-485
5	GND	Ground	RS-232/RX-485
6	Y	TX+	RS-485
7	RTS		RS-232
8	CTS		RS-232
9	B	RX-	RS-485

Pin orientation

1	2	3	4	5
6	7	8	9	

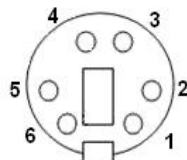
# NeoFox Sport Module

## NeoFox Connectors

### Analog Output Connector

#### Caution

Be careful when wiring. Incorrect wiring could cause permanent damage to the unit.



View of Connector on Back of NeoFox Unit

1 = RS232 Rx  
 2 = 0-5V – V+  
 3 = Current 4-20 - V- (Iout)  
 4 = Current 4-20 - V+  
 5 = RS232 Tx  
 6 = 0-5V / RS232 GND

***This is the view looking at the connector on the back of the NeoFox, with power plug on the right and the USB plug on the left. The connector is a standard PS-2 connector.***

See the [NeoFox and NeoFox Sport Installation and Operation Manual](#) for more information.

# Appendix A

## JAZ-INDY Serial Port Interface Communications and Control Information

### Overview

This appendix describes the software interface provided by the Jaz Serial Bus Application API. The Serial Bus API is designed to communicate with the host (typically an OEM host) through a full duplex, serial port. Currently, it works with the DPU-GPIO serial port (RS232). It is also available with other serial ports that may be added to the Jaz, such as the Industrial Control Module's RS232 or RS485 devices.

### Programming Suggestions

- Send the following sequence when starting your program:
  1. Flush the Jaz Serial Command input buffer
  2. Disable ETX response
  3. Set protocol to Binary
- When using ASCII protocol, CR/LF will only be included if a data field exists in a Jaz response.
- Compression only affects the spectral data portion of the Request Spectrum command.
- The Request Spectrum command always returns a 32-bit pixel.
- Enabling the ETX response is recommended only for ASCII protocol.

# Command Summary

Binary	ASCII	Description	Version
0x3F	?	Parameter Value Query	1.00.0
0x41	A	Set Scan to Average (Add Scans)	1.00.0
0x42	B	Set Pixel Boxcar Width	1.00.0
0x47	G	Set Data Compression	1.00.0
0x49	I	Set Integration Time (msec)	1.00.0
0x4A	J	Set Lamp Enable	1.00.0
0x4B	K	Set Baud Rate	1.00.0
0x50	P	Set Pixel Mode	1.00.0
0x53	S	Start Spectra Acquisition	1.00.0
0x54	T	Set Trigger Mode	1.00.0
0x57	W	Set FPGA Register	Note 1
0x61	a	Set ASCII Mode for Data Values	1.00.0
0x61	b	Set Binary Modes for Data Values	1.00.0
0x69	i	Set Integration Time (µsec)	1.00.0
0x6B	k	Set Checksum Mode	Note 2
0x6C	l	Set Lamp Module Access	1.00.0
0x73	s	Set Spectrometer Module Address	1.00.0
0x76	v	Version Number Query	1.00.0
0x78	x	Set Calibration Coefficients	1.00.0
Note 1: Not currently implemented			
Note 2: Although this command will return an ACK, it is not currently included in the spectral data			

## Command Descriptions

A detailed description of the serial 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.

### Flush Serial Port Input

Causes the JAZ-INDY to flush the contents of the input buffer.

Request	Response
ASCII / Binary: 0x1B 0x1B 0x1B 0x1B [3 bytes]	ASCII / Binary: No Response

## Parameter Value Query

Sets the number of discrete spectra to be summed together. Since Jaz has the ability to return 32-bit values, overflow of the raw 16-bit ADC is not a concern.

	ASCII	Binary
Command Example:	?A<CR>	0x3F 0x0041
Value:	A – scans to average (default = 1) B – pixel boxcar width (default = 0) G – data compression (default = 0) I – integration time (msec) (default = 100) J – lamp enable (default = 0) K – baud rate (default = 5) P – pixel mode (default = 0) T – trigger mode (default = 0) i – integration time (µsec) (default = 100,000) k – checksum (default = 0) l – lamp channel (default = 0) s – spectrometer channel (default = 0) x – calibration index	
Response (Valid Query)	See following table	
Response (Invalid Query) Length:	<NAK> 1 byte	
Default value:	N/A	

Value	Request	Response
Scans to Average (A)	<b>ASCII:</b> “?” + “A” + CR [3 bytes] <b>Binary:</b> 0x3F + 0x41 [2 bytes]	<b>ASCII:</b> ACK + {parameter: 1 – 4 numeric digits} + CR + LF + (ETX, if enabled) [min=4, default=4, max=7 bytes (+ ETX)] <b>Binary:</b> ACK + {unsigned 16-bit value} + (ETX, if enabled) [3 bytes (+ ETX)]
Pixel Boxcar Width (B)	<b>ASCII:</b> “?” + “B” + CR [3 bytes] <b>Binary:</b> 0x3F + 0x42 [2 bytes]	<b>ASCII:</b> ACK + {1-5 numeric digits} + CR + LF + (ETX, if enabled) [min=4, default=4, max=9 bytes (+ ETX)] <b>Binary:</b> ACK + {unsigned 16-bit value} + (ETX, if enabled) [3 bytes (+ ETX)]
The following parameters have a single digit in their responses: Data Compression (G) Lamp Enable (J) Baud Rate (K) Trigger Mode (T) Checksum (k) Lamp Channel (l) Spectrometer Channel (s)	<b>ASCII:</b> “?” + {argument: 1 alpha digit} + CR [3 bytes] <b>Binary:</b> 0x3F + {argument: unsigned 8-bit value} [2 bytes]	<b>ASCII:</b> ACK + {1 numeric digit} + CR + LF + (ETX, if enabled) [4 bytes (+ ETX)] <b>Binary:</b> ACK + {unsigned 16-bit value} + (ETX, if enabled) [3 bytes (+ ETX)]
Integration Time mSec (I)	<b>ASCII:</b> “?” + “i” + CR [3 bytes] <b>Binary:</b> 0x3F + 0x69 [2 bytes]	<b>ASCII:</b> ACK + {1-5 numeric digits} + CR + LF + (ETX, if enabled) [min=4, default=6, max=8 bytes (+ ETX)] <b>Binary:</b> ACK + {unsigned 16-bit value} + (ETX, if enabled) [3 bytes (+ ETX)]
Integration Time µSec (i)	<b>ASCII:</b> “?” + “i” + CR [3 bytes] <b>Binary:</b> 0x3F + 0x69 [2 bytes]	<b>ASCII:</b> ACK + {parameter: 4 – 8 numeric digits} + CR + LF + (ETX, if enabled) [min=7, default=9, max=11 bytes (+ ETX)] <b>Binary:</b> ACK + {unsigned 32-bit value} + (ETX, if enabled) [5 bytes (+ ETX)]

Value	Request	Response
Pixel Mode (P)	<b>ASCII:</b> “?” + “P” + CR [3 bytes] <b>Binary:</b> 0x3F + 0x50 [2 bytes]	<b>ASCII:</b> Pixel Mode 0: ACK + “0” + CR + LF + (ETX, if enabled) [4 bytes (+ ETX)] Pixel Mode 1: ACK + “1” + CR + LF + {Nth pixel element: 1 – 4 numeric digits} + CR + LF + (ETX, if enabled) [min=7, max=10 bytes (+ ETX)] Pixel Mode 3: ACK + “3” + CR + LF + {start pixel: 1 – 4 numeric digits} + CR + LF + {end pixel: 1 – 4 numeric digits} + CR + LF + {Nth pixel element: 1 – 4 numeric digits} + CR + LF + (ETX, if enabled) [min=13, max=22 bytes (+ ETX)] Pixel Mode 4: ACK + “4” + CR + LF + {# of pixels (up to 10): 1 – 2 numeric digits} + CR + LF + {1st pixel: 1 – 4 numeric digits} + CR + LF .... {Nth pixel element: 1 – 4 numeric digits} + CR + LF + (ETX, if enabled) [10-13(1), 1320(2), 16-26(3), 19-??(4), 22-??(5), 25-??(6), 28-??(7), 31-??(8), 34-??(9), 56(10) bytes (+ ETX)]  <b>Binary:</b> Pixel Mode 0: ACK + 0x0000 + (ETX, if enabled) [3 bytes (+ ETX)] Pixel Mode 1: ACK + 0x0001 + {Nth pixel element: unsigned 16-bit value} + (ETX, if enabled) [5 bytes (+ ETX)] Pixel Mode 3: ACK + 0x0003 + {start pixel: unsigned 16-bit value} + {end pixel element: unsigned 16-bit value} + {Nth pixel element: unsigned 16-bit value} + (ETX, if enabled) [9 bytes (+ ETX)] Pixel Mode 4: ACK + 0x0004 + {# of pixels (# <= 10): unsigned 16-bit value} + {1st pixel element: unsigned 16-bit value} + .... + {Nth pixel element: unsigned 16-bit value} + (ETX, if enabled) [7(1), 9(2), 11(3), 13(4), 15(5), 17(6), 19(7), 21(8), 23(9), 25(10) bytes (+ ETX)]
Calibration Index (x)	<b>ASCII:</b> “?” + “x” + {slot #: 1 numeric digit} + CR [4 bytes] <b>Binary:</b> 0x3F + 0x78 + {slot #: unsigned 16-bit value} [4 bytes]	<b>ASCII/Binary:</b> ACK + {coefficient-info: 15 items (unsigned 8-bit value each)} + (ETX, if enabled) [16 bytes (+ ETX)]



### Set Scans to Average (Add Scans)

Sets the number of discrete spectra to be summed together. Since Jaz has the ability to return 32-bit values, overflow of the raw 16-bit ADC is not a concern.

	ASCII	Binary
Command Example:	"A5000"<CR>	0x41 0x1388
Value:	1 – 5000	
Default value:	1	
Response (Valid Query)	See following table	
Response (Invalid Query) Length:	<NAK> 1 byte	

Request	Response
<b>ASCII:</b> "A" + {argument: 1-4 numeric digits} + CR [min=3, max=6 bytes] <b>Binary:</b> 0x41 + {argument: unsigned 16-bit value} [3 bytes]	<b>ASCII/Binary:</b> ACK + (ETX, if enabled) [1 byte (+ ETX)]

### Set Pixel Boxcar Width

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 ~3 will exceed the idle time between values and slow down the overall transfer process.

	ASCII	Binary
Command Example:	"B15"<CR>	0x42 0x000F
Value:	0 – 15	
Default value:	0	
Response (Valid Query)	See following table	
Response (Invalid Query) Length:	<NAK> 1 byte	

Request	Response
<b>ASCII:</b> "B" + {argument: 1-2 numeric digits} + CR [min=3, max=4 bytes] <b>Binary:</b> 0x42 + {argument: unsigned 16-bit value} [3 bytes]	<b>ASCII/Binary:</b> ACK + (ETX, if enabled) [1 byte (+ ETX)]

## Set Data Compression

Specifies whether the data transmitted from Jaz should be compressed to reduce data transfer times.

	ASCII	Binary
Command Example:	"G1"<CR>	0x47 0x0001
Value:	0 = Compression off 1 = Compression on	
Default value:	0	
Response (Valid Query)	See following table	
Response (Invalid Query) Length:	<NAK> 1 byte	

Request	Response
<b>ASCII:</b> "G" + {argument: 1 numeric digit} + CR [3 bytes] <b>Binary:</b> 0x47 + {argument: unsigned 16-bit value} [3 bytes]	<b>ASCII/Binary:</b> ACK + (ETX, if enabled) [1 byte (+ ETX)]

## Set Integration Time (mSec)

Sets the Jaz integration time, in milliseconds, to the value specified.

	ASCII	Binary
Command Example:	"I65535"	0x49 0xFFFF
Value:	1 – 65535	
Default value:	10	
Response (Valid Query)	See following table	
Response (Invalid Query) Length:	<NAK> 1 byte	

Request	Response
<b>ASCII:</b> "I" + {argument: 1-5 numeric digits} + CR [min=3, max=7 bytes] <b>Binary:</b> 0x49 + {argument: unsigned 16-bit value} [3 bytes]	<b>ASCII/Binary:</b> ACK + (ETX, if enabled) [1 byte (+ ETX)]

## Set Lamp Enable

Sets Jaz Lamp Enable output (GPIO line) to the value specified.

	ASCII	Binary
Command Example:	"J0" <CR>	0x4A 0x0030
Value:	0 = Light source/strobe off—Lamp Enable low 1 = Light source/strobe on—Lamp Enable high	
Default value:	0	
Response (Valid Query)	See following table	
Response (Invalid Query) Length:	<NAK> 1 byte	

Request	Response
<b>ASCII:</b> "J" + {argument: 1 numeric digit} + CR [3 bytes] <b>Binary:</b> 0x4A + {argument: unsigned 16-bit value} [3 bytes]	<b>ASCII/Binary:</b> ACK + (ETX, if enabled) [1 byte (+ ETX)]

## Set Baud Rate

Sets Jaz baud rate.

	ASCII	Binary
Command Example:	"K6" <CR>	0x4B 0x0036
Value:	0 (2400) 1 (4800) 2 (9600) 3 (19200) 4 (38400) 5 (57600) 6 (115200)	
Default value:	5 (57600)	
Response (Valid Query)	See following table	
Response (Invalid Query) Length:	<NAK> 1 byte	

Request	Response
<b>ASCII:</b> "J" + {argument: 1 numeric digit} + CR [3 bytes] <b>Binary:</b> 0x4A + {argument: unsigned 16-bit value} [3 bytes]	<b>ASCII/Binary:</b> ACK + (ETX, if enabled) [1 byte (+ ETX)]

## Set Pixel Mode

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

	ASCII	Binary
Command Example:	"P0" <CR>	0x50 0x0000
Value:	Description 0 = all 2048 pixels 1 = every n <sup>th</sup> 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)	
Default value:	0	
Response (Valid Query)	See following table	
Response (Invalid Query) Length:	<NAK> 1 byte	

Request	Response
<b>ASCII:</b> Pixel Mode 0: "P" + "0" + CR + (ETX, if enabled) [3 bytes] Pixel Mode 1: "P" + "1" + CR + {Nth pixel element: 1 – 4 numeric digits} + CR [min=5, max=8 bytes] Pixel Mode 3: "P" + "3" + CR + {start pixel: 1 – 4 numeric digits} + CR + {end pixel: 1 – 4 numeric digits} + CR + {Nth pixel element: 1 – 4 numeric digits} + CR [min=9, max=18 bytes] Pixel Mode 4: "P" + "4" + CR + {# of pixels (up to 10): 1 – 2 numeric digits} + CR + {1st pixel: 1 – 4 numeric digits} + CR .... {Nth pixel element: 1 – 4 numeric digits} + CR [7-10(1), 9-15(2), 11-20(3), 13-25(4), 15-??(5), 17-??(6), 19-??(7), 21-??(8), 23-??(9), 26-56(10) bytes] <b>Binary:</b> Pixel Mode 0: 0x50 + 0x0000 [3 bytes] Pixel Mode 1: 0x50 + 0x0001 + {Nth pixel element: unsigned 16-bit value} [5 bytes] Pixel Mode 3: 0x50 + 0x0003 + {start pixel: unsigned 16-bit value} + {end pixel element: unsigned 16-bit value} + {Nth pixel element: unsigned 16-bit value} [9 bytes] Pixel Mode 4: 0x50 + 0x0004 + {# of pixels (# <= 10): unsigned 16-bit value} + {1st pixel element: unsigned 16-bit value} + .... + {Nth pixel element: unsigned 16-bit value} [7(1), 9(2), 11(3), 13(4), 15(5), 17(6), 19(7), 21(8), 23(9), 25(10) bytes]	<b>ASCII/Binary:</b> ACK + (ETX, if enabled) [1 byte (+ ETX)]

## Start Spectra Acquisition

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. An STX (ASCII 2) is sent once the data is acquired and stored. The format of returned spectra includes a header to indicate scan number, channel number, pixel mode, etc.

	ASCII	Binary
Command Example:	"S"	0x53
Associated Operating Parameters:	Set Scans to Average (Add Scans) (A) (see p. 41) Set Pixel Boxcar Width (B) (see p. 44) Set Trigger Mode (T) (see p. 46) Set Integration Time mSec (I) (see p. 42) Set Lamp Enable (J) (see p.43)	

### Notes:

- The format of returned spectra includes a 13-byte header to indicate the following:  
STX character (1 byte): 0x02  
Prefix Constant (2 bytes): ASCII 65535; Binary 0xFFFF  
Data Size Flag (2 bytes): 1 = pixel data given in 32-bit values  
Number of Scans To Add (2 bytes)  
Integration Time (mSec) (2 bytes)  
FPGA established baseline value (2 bytes): ASCII 0; Binary 0x0000  
Pixel Mode (2 bytes)
- Pixel Data: 2048 pixels (8192 bytes)
- End of Spectrum Flag (2 bytes): ASCII 65533; Binary 0xFFFD
- ASCII (only available when no compression used): STX + {Prefix constant: 5 numeric digits} + CR + LF + {Header: 5 items (1-5 numeric digits + CR + LF each)} + {spectra: 2048 pixels (1-5 numeric digits + CR + LF each)} + {end-of-spectra constant: 5 numeric digits} + CR + LF + (ETX, if enabled) [min=6174, max=14381 bytes (+ ETX)]
- Binary: STX + {prefix constant: unsigned 16-bit value} + {header: 5 items (unsigned 16-bit value)} + {spectra: 2048 pixels (unsigned 32-bit value)} + {end-of-spectra constant: unsigned 16-bit value} + (ETX, if enabled) [8207 bytes (+ ETX)]

Request	Response
<b>ASCII:</b> "S" [1 byte] <b>Binary:</b> 0x53 + {argument: unsigned 16-bit value} [3 bytes]	<b>ASCII</b> (only available when no compression used): STX + {Prefix constant: 5 numeric digits} + CR + LF + {Header: 5 items (1-5 numeric digits + CR + LF each)} + {spectra: 2048 pixels (1-5 numeric digits + CR + LF each)} + {end-of-spectra constant: 5 numeric digits} + CR + LF + (ETX, if enabled) [min=6174, max=14381 bytes (+ ETX)] <b>Binary:</b> STX + {prefix constant: unsigned 16-bit value} + {header: 5 items (unsigned 16-bit value)} + {spectra: 2048 pixels (unsigned 32-bit value)} + {end-of-spectra constant: unsigned 16-bit value} + (ETX, if enabled) [8207 bytes (+ ETX)]

### Set Trigger Mode

Jaz supports four trigger modes, which are set with the Trigger Mode command.

	ASCII	Binary
Command Example:	"T0" <CR>	0x54 0x0000
Value:	0 = Normal – Continuously scanning 1 = Software trigger 2 = External synchronization 3 = External hardware trigger	
Default value:	0 (Normal – Continuously scanning)	
Response (Valid Query)	See following table	
Response (Invalid Query) Length:	<NAK> 1 byte	

Request	Response
<b>ASCII:</b> "T" + {argument: 1 numeric digit} + CR [3 bytes] <b>Binary:</b> 0x54 + {argument: unsigned 16-bit value} [3 bytes]	<b>ASCII/Binary:</b> ACK + (ETX, if enabled) [1 byte (+ ETX)]

### Set FPGA Register Value

Sets the appropriate register within the FPGA. This command is currently not implemented and will always return a NAK.

## Set ASCII Mode for Data Values

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).

	ASCII	Binary
Command Example:	"aA"	0x61, 0x41
Value:	None	
Response (Valid Query)	See following table	
Response (Invalid Query) Length:	<NAK> 1 byte	

Request	Response
<b>ASCII:</b> "a" + {argument: 1 alpha character} [2 bytes] <b>Binary:</b> 0x61 + {argument: unsigned 8-bit value} [3 bytes]	<b>ASCII/Binary:</b> ACK + (ETX, if enabled) [1 byte (+ ETX)]

## Set Binary Mode for Data Values

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.

	ASCII	Binary
Command Example:	"bB"	0x62, 0x42
Value:	None	
Response (Valid Query)	See following table	
Response (Invalid Query) Length:	<NAK> 1 byte	

Request	Response
<b>ASCII:</b> "b" + {argument: 1 alpha character} [2 bytes] <b>Binary:</b> 0x61 + {argument: unsigned 8-bit value} [2 bytes]	<b>ASCII/Binary:</b> ACK + (ETX, if enabled) [1 byte (+ ETX)]

## Set ETX Response

	ASCII	Binary
Command Example:	“e0” <CR>	0x65 0x01
Value:	0 = Don't include ETX at end of response 1 = Include ETX at end of response	
Response (Valid Query)	See following table	
Response (Invalid Query) Length:	<NAK> 1 byte	
Note: A request to enable ETX will cause an ETX to be included in the response		

Request	Response
<b>ASCII:</b> "e" + {argument: 1 numeric digit} + CR [3 bytes] <b>Binary:</b> 0x65 + {argument: unsigned 8-bit value} [2 bytes]	<b>ASCII/Binary:</b> ACK + (ETX, if enabled) [1 byte (+ ETX)]

## Set Integration Time (µSec)

Sets the Jaz integration time, in microseconds, to the value specified.

	ASCII	Binary
Command Example:	"i65535000" <CR>	0x69 0x3E7FC18
Value:	1000 – 65,535,000	
Response (Valid Query)	See following table	
Response (Invalid Query) Length:	<NAK> 1 byte	

Request	Response
<b>ASCII:</b> "i" + {4-8 numeric digits} + CR [min=6, max=10 bytes] <b>Binary:</b> 0x69 + {unsigned 32-bit value} [3 bytes]	<b>ASCII/Binary:</b> ACK + (ETX, if enabled) [1 byte (+ ETX)]

## Set Checksum Mode

Specifies whether Jaz 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; its use is recommended when reliable data scans are required.



**NOTE:** Even though this command will return an ACK, a checksum is not currently included in the spectral data.

	ASCII	Binary
Command Example:	"k1" <CR>	0x76 0x0001
Value:	0 = Do not transmit checksum value 1 = Transmit checksum value at end of scan	
Response (Valid Query)	See following table	
Response (Invalid Query) Length:	<NAK> 1 byte	

Request	Response
<b>ASCII:</b> "k" + {argument: 1 numeric digit} + CR [3 bytes] <b>Binary:</b> 0x6B + {argument: unsigned 16-bit value} [3 bytes]	<b>ASCII/Binary:</b> ACK + (ETX, if enabled) [1 byte (+ ETX)]

### Set Lamp Module Address

Specifies the index of the lamp module to be used.

**NOTE:** The Lamp Module Address is reserved for future use.

	ASCII	Binary
Command Example:	"I2" <CR>	0x6C 0x0002
Value:	0 – 15	
Default value:	0	
Response (Valid Query)	See following table	
Response (Invalid Query) Length:	<NAK> 1 byte	

Request	Response
<b>ASCII:</b> "I" + {argument: 1-2 numeric digit} + CR [min = 3 bytes; max = 4 bytes] <b>Binary:</b> 0x6C + {argument: unsigned 16-bit value} [3 bytes]	<b>ASCII/Binary:</b> ACK + (ETX, if enabled) [1 byte (+ ETX)]

## Set Spectrometer Module Address

Specifies the index of the spectrometer module to be used.

	ASCII	Binary
Command Example:	"s2" <CR>	0x73 0x0002
Value:	0 – 15	
Default value:	0	
Response (Valid Query)	See following table	
Response (Invalid Query) Length:	<NAK> 1 byte	

Request	Response
<b>ASCII:</b> "s" + {argument: 1-2 numeric digit} + CR [min = 3 bytes; max = 4 bytes] <b>Binary:</b> 0x73 + {argument: unsigned 16-bit value} [3 bytes]	<b>ASCII/Binary:</b> ACK + (ETX, if enabled) [1 byte (+ ETX)]

## Version Number Query

Returns the version number of the serial bus API running on Jaz.

	ASCII	Binary
Command Example:	"v"	0x76
Value:	none	
Response (Valid Query)	See following table	
Response (Invalid Query) Length:	<NAK> 1 byte	

Request	Response
<b>ASCII:</b> "v" [1 byte] <b>Binary:</b> 0x76 [1 byte]	<b>ASCII:</b> ACK + {4 numeric digits} + CR + LF + (ETX, if enabled) [7 bytes (+ ETX)] <b>Binary:</b> ACK + {unsigned 16-bit value} + (ETX, if enabled) [3 bytes (+ ETX)]
Note: Returned version data is interpreted as shown in the following examples: ASCII: "1000" := 1.00.0 Binary : 0x03E8 (1000) := 1.00.0 (big-endian)	

## Set Calibration Constants

Writes one of the 16 possible calibration constants to EEPROM (see Appendix B: [JAZ-INDY Spectrometer EEPROM Information Slots](#) for more information on the EEPROM slots). A calibration constant is specified by the first DATA WORD following the x. The calibration constant is stored as an ASCII string with a maximum length of 15 characters. The string is not checked for validity.

	ASCII	Binary
Command Example:	"x"	0x78
Value:	Slot number 1 – 17	
Response (Valid Query)	See following table	
Response (Invalid Query) Length:	<NAK> 1 byte	

### Data Byte - Description

- 0 – Serial Number
- 1 – 0<sup>th</sup> order Wavelength Calibration Coefficient
- 2 – 1<sup>st</sup> order Wavelength Calibration Coefficient
- 3 – 2<sup>nd</sup> order Wavelength Calibration Coefficient
- 4 – 3<sup>rd</sup> order Wavelength Calibration Coefficient
- 5 – Stray light constant
- 6 – 0<sup>th</sup> order non-linearity correction coefficient
- 7 – 1<sup>st</sup> order non-linearity correction coefficient
- 8 – 2<sup>nd</sup> order non-linearity correction coefficient
- 9 – 3<sup>rd</sup> order non-linearity correction coefficient
- 10 – 4<sup>th</sup> order non-linearity correction coefficient
- 11 – 5<sup>th</sup> order non-linearity correction coefficient
- 12 – 6<sup>th</sup> order non-linearity correction coefficient
- 13 – 7<sup>th</sup> order non-linearity correction coefficient
- 14 – Polynomial order of non-linearity calibration
- 15 – Optical bench configuration: gg fff sss  
gg – Grating #, fff – filter wavelength, sss – slit size
- 16 – Jaz configuration: AWL V  
A – Array coating Mfg, W – Array wavelength (VIS, UV, OFLV), L – L2 lens installed,  
V – CPLD Version
- 17 – Autonulling
- 18 – Reserved
- 19 – Reserved

Request	Response
<b>ASCII:</b> "x" + {slot #: 1-2 numeric digits} + {15 items (unsigned 8-bit value each)} [min=17, max=18 bytes] <b>Binary:</b> 0x78 + {slot #: unsigned 16-bit value} + {15 items (unsigned 8-bit value each)} [18 bytes]	<b>ASCII/Binary:</b> ACK + (ETX, if enabled) [1 byte (+ ETX)]

# Appendix B

## JAZ-INDY Spectrometer EEPROM Information Slots

Slot #	Description	Example
00	Serial Number	"JAZA0429" := {4A 41 5A 41 30 34 32 39 00 07 00 2B 00 20 00}
01	Wavelength Calibration Intercept	"1.7859120e+002" := {31 37 38 2E 35 39 31 32 30 30 00 00 D0 85 A4}
02	Wavelength Calibration First Coefficient	"3.7593100e-001" := {30 2E 33 37 35 39 33 31 00 30 00 00 D0 85 A4}
03	Wavelength Calibration Second Coefficient	"-1.1561300e-001" := {2D 31 2E 31 35 36 31 33 30 65 2D 30 30 35 00}
04	Wavelength Calibration Third Coefficient	"-2.6288800e-009" := {2d 32 2e 36 32 38 38 38 30 65 2D 30 30 39 00}
05	Stray Light Correction Coefficient	"0.0000000e+000" := {30 2e 30 30 30 30 30 30 65 2B 30 30 30 00 00}
06	Nonlinearity Correction 0 <sup>th</sup> Order Coefficient	"0.0000000e+000" := {30 2e 30 30 30 30 30 30 65 2B 30 30 30 00 00}
07	Nonlinearity Correction 1 <sup>st</sup> Order Coefficient	"0.0000000e+000" := {30 2e 30 30 30 30 30 30 65 2B 30 30 30 00 00}
08	Nonlinearity Correction 2 <sup>nd</sup> Order Coefficient	"0.0000000e+000" := {30 2e 30 30 30 30 30 30 65 2B 30 30 30 00 00}
09	Nonlinearity Correction 3 <sup>rd</sup> Order Coefficient	"0.0000000e+000" := {30 2e 30 30 30 30 30 30 65 2B 30 30 30 00 00}
10	Nonlinearity Correction 4 <sup>th</sup> Order Coefficient	"0.0000000e+000" := {30 2e 30 30 30 30 30 30 65 2B 30 30 30 00 00}
11	Nonlinearity Correction 5 <sup>th</sup> Order Coefficient	"0.0000000e+000" := {30 2e 30 30 30 30 30 30 65 2B 30 30 30 00 00}
12	Nonlinearity Correction 6 <sup>th</sup> Order Coefficient	"0.0000000e+000" := {30 2e 30 30 30 30 30 30 65 2B 30 30 30 00 00}
13	Nonlinearity Correction 7 <sup>th</sup> Order Coefficient	"0.0000000e+000" := {30 2e 30 30 30 30 30 30 65 2B 30 30 30 00 00}
14	Nonlinearity Correction Order	"0" := {30 00 30 30 30 30 30 30 65 2B 30 30 30 00 00}
15	Configuration String 1	"02 000 025" := {30 32 20 30 30 30 20 30 32 35 00 FF FF FF FF}
16	Configuration String 2	"B41 A" := {42 34 31 20 41 00 FF FF FF FF FF FF FF FF}
17	Autonulling Info <i>Note: This slot is shown for informational purposes only. The autonulling feature is built into the spectral data output.</i>	"Enabled,Disabled,13400,29200" := {03 00 58 34 10 72 FF FF FF FF FF FF FF FF} Enabled,Disabled := 0x03 13400 := 0x5834 (little-endian) 29200 := 0x1072 (little-endian)