PX4 PROGRAMMER'S GUIDE

1 OVERVIEW

Amptek's PX4 is a Digital Pulse Processor/Multi-channel Analyzer that is optimized for use with the Amptek XR100 X-Ray detectors. The PX4 includes low- and high-voltage supplies, a supply for operating the XR100's integrated solid-state cooler, an RS232 interface, a USB1.1 interface, and various inputs and outputs.

Amptek provides a DLL to make software development under Windows fairly painless. This document describes how to communicate directly with the PX4 via its RS232 and USB interfaces, rather than using the DLL. The DLL is documented separately. [Currently, the DLL only supports the USB interface.]

2 RS232 SERIAL INTERFACE

The PX4 uses the RS232 at 57.6kbaud, with 1 start bit, 8 data bits, 1 stop bit and no parity. No hardware or software handshaking is used (i.e. no RTS/CTS or XON/XOFF, etc.). Note that there are no checksums or other checking.

The PX4 uses a 'gap timer' to aid in synchronization. If it detects an inter-byte gap of more than approximately 44mS, it discards the received data and starts looking for the sync byte.

2.1 RS232 PACKET TYPES

All communications are initiated by the host computer; the PX4 only sends data in response to the reception of a packet from the host. There are three types of packets the PX4 recognizes: a configuration packet, a data request packet, and a function request packet. The PX4 will respond to a configuration packet by immediately updating its configuration. It will respond to a data request packet by sending 256 bytes of data. It immediately executes a function request but does not send a response.

Every packet sent to the PX4 begins with a synchronization byte (0xFD), and ends with a byte indicating what type of packet it is – a configuration packet (0xFE) or a data/function request packet (0xFF).

2.1.1 RS232 Configuration Packet

The configuration packet via RS232 consists of the 64-byte USB configuration packet (described in section 3.1.2), preceded by a sync byte, and followed by an end byte, for a total of 66 bytes.

Byte 1	0xFD (sync)
Bytes 2-65	64 configuration bytes, identical to USB configuration packet (see Section 3.1.2)
Byte 66	0xFE (end of configuration packet)

Upon reception of the configuration packet, the PX4 will immediately use the new configuration.

2.1.2 RS232 Data Request Packet

Data request packet (3 bytes):

Byte 1	0xFD (sync)
Byte 2	requested packet number
Byte 3	0xFF (end of packet request)

The PX4 will respond to a data request by sending 256 bytes of data.

MCA Mode	Buffer A Packet Numbers	Buffer B Packet Numbers
256 channels	0x00-0x02 (3 packets)	0x80-0x82 (3 packets)

512 channels	0x00-0x05 (6 packets)	0x80-0x85 (6 packets)
1024 channels	0x00-0x0B (12 packets)	0x80-0x8B (12 packets)
2048 channels	0x00-0x17 (24 packets)	0x80-0x97 (24 packets)
4096 channels	0x00-0x2F (48 packets)	0x80-0xAF (48 packets)
8192 channels	0x00-0x5F (96 packets)	0x80-0xDF (96 packets)
Status packet	0x60	0xE0
Configuration readback	0x61	(same as Buffer A)
Oscilloscope packets	0x62-0x63	(same as Buffer A)
Status packet plus Clear spectrum[1]	0x64	0xE4

2.1.3 RS232 Data Response Formats

There are 4 different types of data packets that are transmitted in response to a Data Request: spectral, status, configuration, and oscilloscope data. All of these consist of one or more 256-byte packets.

2.1.3.1 Spectral Data

The spectral data format is 3 bytes/channel, in order from LSB to MSB, and in channel number from lowest to highest. Note that at 3 bytes per channel, an integral number of channels doesn't fit in the 256 byte packet, and therefore some channels may span packet boundaries. For example, packet #0x00 contains the 3-byte channel data for channels 1 through 85, followed by the first byte (LSB) of channel 86. Packet #0x01 starts with the two higher-order bytes of channel 86, followed by channels 87 through 170, followed by the two lower-order bytes of channel 171, etc. For this reason, it is recommended that the entire spectrum be read out, and the spectral packets should be assembled in order into one buffer. [This is a recommendation, not a requirement. There is no requirement that the entire spectrum, or indeed any of the spectrum, be read out. It is possible to read only the packets which contain data of interest, just the status data, or no data at all.]

For the MCS (Multi Channel Scaler) mode, the data is in the same format except each bin is a time bin rather than an energy bin. The output of SCA8 is the source of counts, so it must be enabled and the thresholds set.

2.1.3.2 Status Data

Status data consists of 64 bytes of status data, identical to the USB status packet (see section 3.1.3), followed by 192 bytes of zeroes (0x00) to fill the 256-byte packet.

2.1.3.3 Configuration Readback Data

Configuration readback data consists of 64 bytes of configuration data, identical to the USB configuration packet (see section 3.1.2), followed by 192 bytes of zeroes (0x00) to fill the 256-byte packet. The configuration readback data reflects the current configuration of the instrument – it contains the configuration information from the most recently received configuration packet. If no configuration packet has been received, it returns the configuration stored in the PX4's non-volatile memory. [Status packet offset 23, bits D1-D0 indicate whether the PX4 is configured or not, and whether its current configuration came from its non-volatile memory via a 'power button configuration'.]

2.1.3.4 Oscilloscope data

The oscilloscope data consists of two 256-byte packets, for a total of 512 bytes. The data is a reduced-resolution version of what is sent to the output DAC – the oscilloscope data uses 8 bits per sample. The

[1] 'Status Packet plus Clear Spectrum' data request requires firmware v4.01 or later.

timebase is automatically selected internally – if the 'FAST' signal is selected for the DAC, then the internal oscilloscope uses the full 20MHz clock (i.e. 50nS per sample). Otherwise, the internal oscilloscope uses the decimated clock as the timebase, as shown in the following table:

Peaking Time	**Internal Oscilloscope Timebase**
0.8 – 6.4uS	50nS per sample
8.0 – 12.8uS	100nS per sample
16.0 – 25.6uS	200nS per sample
32.0 – 51.2uS	400nS per sample
64.0 – 102.4uS	800nS per sample

2.1.4 RS232 Function Request Packet

Function request packet (3 bytes):

Byte 1	0xFD (sync)
Byte 2	Function request number (0x70-0x77)
Byte 3	0xFF (end of packet request)

No data is sent by the PX4 in response to a function request.

RS232 Function Request	**Function**
0x70	Clear spectrum buffer A
0x71	Clear spectrum buffer B
0x72	Disable MCA/MCS
0x73	Enable MCA/MCS
0x74	Arm digital oscilloscope
0x75	Autoset Input Offset (fine steps)[2,3]
0x76	Autoset Input Offset (coarse steps followed by fine steps)[2,3]
0x77	Autoset fast threshold[4]

2.2 TYPICAL RS232 SEQUENCE

Typical RS232 communications sequence:

Send a 66-byte configuration packet to configure the DP4 and start taking data.

[2] 'Autoset Input Offset' only searches in one direction. Therefore, the initial Input Offset should be the most negative setting (-2.048V) if the analog front end is set to 'Inverting', or the most positive setting (+2.047V) if the front end is set to 'non-inverting'.

[3] In general, the 'coarse steps' function (0x76) is preferable to the 'fine steps' function. The 'fine steps' function may be faster if the starting input offset is close to the correct setting, but it can take an inordinate amount of time if it isn't close.

[4] 'Autoset fast threshold' only works with no source present and low levels of background counts (less than approx. 20 cps.)

Periodically (once per second, for example):
Request spectrum data
Request status data

Here is an example of how to request spectrum data, using 512 channel operation as an example:

1. Send: 0xFD 0x00 0xFF ('Request Data', Packet Number=0, 'Sync')
2. Receive 256 bytes of data
3. Send: 0xFD 0x01 0xFF ('Request Data', Packet Number=1, 'Sync')
4. Receive 256 bytes of data
5. Send: 0xFD 0x02 0xFF ('Request Data', Packet Number=2, 'Sync')
6. Receive 256 bytes of data
7. Send: 0xFD 0x03 0xFF ('Request Data', Packet Number=3, 'Sync')
8. Receive 256 bytes of data
9. Send: 0xFD 0x04 0xFF ('Request Data', Packet Number=4, 'Sync')
10. Receive 256 bytes of data
11. Send: 0xFD 0x05 0xFF ('Request Data', Packet Number=5, 'Sync')
12. Receive 256 bytes of data

Read the status either of two ways. This method allows the data taking to continue:
13. Send: 0xFD 0x60 0xFF ('Request Data', Packet Number=Status, 'Sync')
14. Receive 256 bytes of status

This method clears the spectrum and status data - in other words, it restarts the accumulation:
13. Send: 0xFD 0x64 0xFF ('Request Data', Packet Number=Status+Clear Data, 'Sync')
14. Receive 256 bytes of status

Read the configuration packet (if desired)
15. Send: 0xFD 0x61 0xFF ('Request Data', Packet Number=Config readback, 'Sync')
16. Receive 256 bytes of status

If an 'Arm digital oscilloscope' function request was sent previously, check the status packet, offset 23d bit D2 to see if oscilloscope data is ready (bit D2=1). If so,
17. Send: 0xFD 0x62 0xFF ('Request Data', Packet Number=1st Oscilloscope packet, 'Sync')
18. Receive 256 bytes of oscilloscope data
19. Send: 0xFD 0x63 0xFF ('Request Data', Packet Number=2nd Oscilloscope packet, 'Sync')
20. Receive 256 bytes of oscilloscope data

How frequently you can request the data depends on how many channels are selected. Here are the approximate times required to transmit the spectrum (3 bytes are used for each spectrum channel):

256 channels+status	~150mS
512 channels+status	~250mS
1024 channels+status	~500mS
2048 channels+status	~1S
4096 channels+status	~2S
8192 channels+status	~3.5S

Note: When the PX4 serial interface is processing a received packet, it may be busy and is unable to process another packet. This is a direct consequence of the PX4 having received a packet, and therefore is very predictable. It takes less than 1 mS for it to process a Data Request packet and to start sending data. It takes approximately 10mS to process a Configuration Packet, and it takes approximately 40mS to clear the spectrum memory (via either a Function Request, or a 'Status packet plus Clear spectrum' Data Request.)

The application software should wait until the data response is received from a Data Request before sending another packet to the PX4. Since the brief RS232 'dead time' occurs between the reception of the request packet and the transmission of the resulting data, the application can avoid this 'dead time' by not sending another Data Request until the data from the previous one has been received. (This includes the 'Status packet plus Clear spectrum' request – the ~40mS of 'dead time' occurs after the request is received, but before the status data is transmitted.)

3 USB INTERFACE

3.1.1 Function Calls

Following are details of the USB function calls in the Visual Basic sample application. Refer to the documentation for the APA USB driver for specifics (available upon request), and to the Visual Basic sample application source code to see how each function is used.

USBDRVD_GetDevCount (called throughout the sample application.)

This is called prior to opening the USB port, to be sure that a PX4 is available. It's also called prior to most PX4 USB transfers, to be sure that the PX4 hasn't been disconnected or powered off.

USBDRVD_OpenDevice (called by *btnConfigure*)

This opens the device for handle access, which is required for USBDRVD_VendorOrClassRequestOut.

USBDRVD_PipeOpen (called by *btnConfigure*)

This opens the various PX4 pipes, which are used by USBDRVD_PipeWriteTimeout and USBDRVD_PipeReadTimeout:

USB Pipe	Pipe Function	Data Size
Pipe 0	Configuration data OUT to PX4 (USB Device Endpoint OUT1)	64 bytes
Pipe 1	Buffer A Status IN from PX4 (USB Device Endpoint IN1)	64 bytes
Pipe 2	Buffer A Spectrum IN from PX4 (USB Device Endpoint IN2)	768,1536...24576 (# of MCA channels x3)
Pipe 3	Buffer B Status IN from PX4 (USB Device Endpoint IN3)	64 bytes
Pipe 4	Buffer B Spectrum IN from PX4 (USB Device Endpoint IN4)	768,1536...24576 (# of MCA channels x3)
Pipe 5	Configuration readback IN from PX4 (USB Device Endpoint IN5)	64 bytes
Pipe 6	Oscilloscope trace data IN from PX4 (USB Device Endpoint IN6)	512 bytes

USBDRVD_PipeWriteTimeout (called by *btnConfigure*)

This is used to send the 64-byte configuration packet to the PX4.

USBDRVD_CloseDevice (called by *btnExit_Click* and *Timer2_Timer*)

This closes the device handle. Timer2 will close the device if it is open but USBDRVD_GetDevCount detects no PX4s, due to disconnection or powering down.

USBDRVD_PipeClose (called by *btnExit_Click* and *Timer2_Timer*)

This closes the open pipes. Timer2 will close the handles if they are open but USBDRVD_GetDevCount detects no PX4s, due to disconnection or powering down.

USBDRVD_VendorOrClassRequestOut (called throughout the sample application.)

This sends a zero-length 'vendor request' packet to the PX4. (The packet format is described in section 3.1.2.)

The btnPauseMCA_Click routine calls it with a request of 0x83 to disable (i.e pause) the MCA or 0x82 to enable (i.e resume) the MCA.

The Timer1_Timer routine calls it after a spectrum transfer to clear spectrum buffer A (request=0x80) or spectrum buffer B (request=0x81).

The SCA Form calls it to send an 'Arm Digital Oscilloscope' in response to the 'Arm Trigger' button.

There are other miscellaneous calls throughout the software.

USB 'Vendor Request'	**Function**
0x7F	Write 'PX4 Name'[5]
0x80	Clear spectrum buffer A
0x81	Clear spectrum buffer B
0x82	Enable MCA/MCS
0x83	Disable MCA/MCS
0x84	Arm digital oscilloscope
0x85	Autoset Input Offset (fine steps)
0x86	Autoset Input Offset (coarse steps followed by fine steps)
0x87	Autoset fast threshold[6]
0x90	Set Diode Calibration DAC (DCAL)[7]
0x91	Set TEC Current Limit DAC (TECCL)[7]

USBDRVD_PipeReadTimeout (called by *Timer1_Timer*)

This is called to retrieve the entire spectrum from either buffer A or buffer B, the status packet for buffer A or buffer B, the current configuration (or the stored configuration if the unit is unconfigured), or oscilloscope trace data. The spectral data format is 3 bytes/channel, in order from LSB to MSB, and in channel number from lowest to highest. The status packet format is described in section 3.1.3.

The oscilloscope trace data is 512 bytes in length.

Note: for proper operation, the calling program should request a byte count of the expected amount plus one. For example, in 1024 channel mode, a spectrum request should ask for 3073 bytes rather than 3072. A status packet request should request 65 bytes, not 64.

[5] The PX4 stores a 64-character string in non-volatile memory, which can be written to with Vendor Request 0x7F, and read by calling 'USBDRVD_GetStringDescriptor'. Vendor Request 0x7F writes two characters at a time; the 'index' field specifies the offset in the string for the two characters (0, 2, 4…62) and the 'value' field holds the two characters – the LSB is the 1st character and the MSB is the 2nd character. This Vendor Request must be sent 32 times to fill the 64-character string – the string will not be written into non-volatile memory if all 32 Requests are not received.

[6] 'Autoset fast threshold' only works with no source present and low levels of background counts (less than approx. 20 cps.)

[7] 'DCAL' and 'TECCL' settings are stored in non-volatile memory and shouldn't need to be changed. The 'value' field of the Vendor Request contains the new DAC setting. Contact Amptek for more information.

3.1.2 USB Configuration Packet Format

Offset	Data bits	Allowed Value	Description
0	D7	0	0=Normal reset lockout, 1=fast reset lockout
	D6-D3	t=0…15	Flat top width=200nS*(t+1)*2^d
	D2-D0	d=0…4	Decimation factor=2^d
1	D7-D0	0…255	Slow channel threshold
2	D7-D0	0…255	Fast channel threshold
3	D7-D1	-64…+63	Output DAC offset (signed) (-500mV to +492mV)
	D0	0	DAC disabled
		1	DAC enabled
4	D7-D6	0	normal operation
	D5	0	MCA disabled
		1	MCA enabled
	D4-D2	0	MCA/MCS 4096 channel mode
		1	MCA/MCS 2048 channel mode
		2	MCA/MCS 1024 channel mode
		3	MCA/MCS 512 channel mode
		4	MCA/MCS 256 channel mode
		5	MCA/MCS 8192 channel mode
	D1-D0	0	DAC output=fast channel
		1	DAC output=shaped pulse
		2	DAC output=decimated input
		3	DAC output= Test Pulser
5	D7-D0	1…255	Pileup Reject interval: optimal setting is (p * 19 + (t + 1) * 4) / 2
		0	Pileup Reject disabled
6	D7-D4	p=1…8	Peaking Time=800nS*p*2^d (See Config byte 0: bits D2-D0, and also Table 2)
	D3-D2	0	Detector reset lockout period=13.11mS (slow) or 819uS (fast) (see Config byte 0: bit D0)
		1	Detector reset lockout period=6.55mS or 410uS
		2	Detector reset lockout period=3.28mS or 205uS
		3	Detector reset lockout period=1.64mS or 102uS
	D1	0	No auto baseline reset during detector reset
		1	Auto baseline reset during detector reset
	D0	0	Disable MCA during detector reset

		1	Do not disable MCA during detector reset
7	D7-D0	0…255	RTD slow threshold
8	D7	1	Normal operation
	D6	0	(Reserved)
	D5	0-1	Analog gain control A (See Table 1.)
	D4	0	RTD off
		1	RTD on
	D3-D0	0…15	RTD time threshold
9	D7	0	50% digital attenuation
		1	Normal operation
	D6	0	0=Baseline Restoration (BLR) off
		1	BLR enabled
	D5-D4	0	BLR Down correction=very slow
		1	BLR Down correction=slow
		2	BLR Down correction=medium
		3	BLR Down correction=fast
	D3-D2	0	BLR Up correction=very slow
		1	BLR Up correction=slow
		2	BLR Up correction=medium
		3	BLR Up correction=fast
	D1-D0	0	BLR correction threshold=very fast
		1	BLR correction threshold=fast
		2	BLR correction threshold=normal
		3	BLR correction threshold=slow
10	D7-D6	0	GATE off
		1	N/A (Gate off)
		2	GATE on, polarity active high
		3	GATE on, polarity active low
	D5-D4	0	Software MCA buffer select, buffer A
		1	Software MCA buffer select, buffer B
		2	Hardware MCA buffer select
		3	N/A
	D3	0	Digital scope trigger = rising edge
		1	Digital scope trigger = falling edge
	D2-D0	0	AUX_OUT=ICR

		1	AUX_OUT=PILEUP
		2	AUX_OUT=MCS_TIMEBASE
		3	AUX_OUT=ONESHOT
		4	AUX_OUT=DET_RES (active low)
		5	AUX_OUT=MCA_EN
		6	AUX_OUT=TRIGGER
		7	AUX_OUT=SCA8
11	D7-D0	0-255	Preset time LSB
12	D7-D0	0-255	Preset time Byte2
13	D7-D0	0-255	Preset time MSB [Time=0.1 sec * [MSB*(2^16)+Byte2*(2^8)+LSB]
14	D7-D6	0	Audible Count Rate Monitor (ACRM) off
		1	ACRM volume=medium
		2	ACRM volume=low
		3	ACRM volume=high
	D5	0	Disable HV power supply
		1	Enable HV power supply
	D4	0	±8.5/5V supply=±5V
		1	±8.5/5V supply=±8.5V
	D3	0	Turn OFF PX4 power supplies
		1	Turn ON PX4 power supplies
	D2	0	Disable ±8.5/5V supply
		1	Enable ±8.5/5V supply
	D1	0	N/A
	D0	0	Disable Thermoelectric Cooler (TEC) supply
		1	Enable TEC supply
15	D7	0	Front-end=non-inverting[8]
		1	Front-end=Inverting[9]
	D6-D4	0	N/A
	D3-D0	0-15	Analog Gain Control B (See Table 1.)
16	D7-D0	0-15	HV MSB
17	D7-D0	0-255	HV LSB

[8] The front-end should be configured as 'Non-inverting' if the input to the PX4 is a positive pulse.

[9] The front-end should be configured as 'Inverting' if the input to the PX4 is a negative pulse or negative-going reset ramp, such as that produced by Amptek's XR100 detectors.

			HV (Volts) =(MSB*256 + LSB) * 0.732V]
18	D7-D0	0-15	TEC Temperature setting MSB
19	D7-D0	0-255	TEC Temperature setting LSB Temp (°C)=[(MSB*256 + LSB)*300/4096]-273
20	D7-D0	0-15	Input Offset MSB
21	D7-D0	0-255	Input Offset LSB Input Offset (mV)=(MSB*256 + LSB)-2048
22	D7-D0	0	Disable Input Pole-Zero
		1-255	Enable input Pole-Zero τ=???
23	D7-D0	0-255	Fine Gain & Normalizer LSB[10]
24	D7-D6	0	Digital scope trigger position=87% pretrigger
		1	Digital scope trigger position=50% pretrigger
		2	Digital scope trigger position=12% pretrigger
		3	Digital scope trigger position=25% delayed trigger
	D5-D0	0-63	Fine Gain & Normalizer MSB
25	D7-D0	0-255	Preset Counts LSB[11]
26	D7-D0	0-255	Preset Counts byte 2
27	D7-D0	0-255	Preset Counts byte 3
28	D7-D0	0-255	Preset Counts MSB [Preset counts=0x0000 if unused]
29	D7	0	MCA mode (see byte 4 for # of channels)
		1	MCS mode (see byte 4 for # of channels)[12]
	D6	0	(Reserved)
	D5	0	MCS disabled
		1	MCS enabled
	D4	0	(Reserved)

[10] 'Fine Gain & Normalizer' setting controls both digital fine gain, and normalizes for different peaking times. The setting is 14 bits in size, and is:

Setting= INT[(FineGain x 8192)/PeakingTime], where

FineGain is in the range 0.75…1.25

PeakingTime is an integer in the range 1…8 (see configuration byte 6 and Table 2)

[11] The 'Preset Count' function counts events produced by SCA8. Therefore, the SCA8 thresholds must be set, and SCA8 must be enabled.

[12] The MCS mode also uses SCA8 as its source of counts. Therefore, the SCA8 thresholds must be set, and SCA8 must be enabled.

	D3-D0	0	MCS Timebase=10mS/channel
		1	MCS Timebase=20mS/channel
		2	MCS Timebase=50mS/channel
		3	MCS Timebase=100mS/channel
		4	MCS Timebase=200mS/channel
		5	MCS Timebase=500mS/channel
		6	MCS Timebase=1 sec/channel
		7	MCS Timebase=2 sec/channel
		8	MCS Timebase=5 sec/channel
		9	MCS Timebase=10 sec/channel
		10	MCS Timebase=20 sec/channel
		11	MCS Timebase=30 sec/channel
		12	MCS Timebase=60 sec/channel
		13	MCS Timebase=90 sec/channel
		14	MCS Timebase=120 sec/channel
		15	MCS Timebase=300 sec/channel
30	D7-D0	0	=0
31	D7-D0	0	=0
32	D7-D0	0-255	SCA1 Lower Threshold LSB
33	D7-D0	0-31	SCA1 Lower Threshold MSB
34	D7-D0	0-255	SCA1 Upper Threshold LSB
35	D7	0	SCA1 disabled
		1	SCA1 enabled
	D6-D0	0-31	SCA1 Upper Threshold MSB
36	D7-D0	0-255	SCA2 Lower Threshold LSB
37	D7-D0	0-31	SCA2 Lower Threshold MSB
38	D7-D0	0-255	SCA2 Upper Threshold LSB
39	D7	0	SCA2 disabled
		1	SCA2 enabled
	D6-D0	0-31	SCA2 Upper Threshold MSB
40	D7-D0	0-255	SCA3 Lower Threshold LSB
41	D7-D0	0-31	SCA3 Lower Threshold MSB
42	D7-D0	0-255	SCA3 Upper Threshold LSB
43	D7	0	SCA3 disabled
		1	SCA3 enabled

	D6-D0	0-31	SCA3 Upper Threshold MSB
44	D7-D0	0-255	SCA4 Lower Threshold LSB
45	D7-D0	0-31	SCA4 Lower Threshold MSB
46	D7-D0	0-255	SCA4 Upper Threshold LSB
47	D7	0	SCA4 disabled
		1	SCA4 enabled
	D6-D0	0-31	SCA4 Upper Threshold MSB
48	D7-D0	0-255	SCA5 Lower Threshold LSB
49	D7-D0	0-31	SCA5 Lower Threshold MSB
50	D7-D0	0-255	SCA5 Upper Threshold LSB
51	D7	0	SCA5 disabled
		1	SCA5 enabled
	D6-D0	0-31	SCA5 Upper Threshold MSB
52	D7-D0	0-255	SCA6 Lower Threshold LSB
53	D7-D0	0-31	SCA6 Lower Threshold MSB
54	D7-D0	0-255	SCA6 Upper Threshold LSB
55	D7	0	SCA6 disabled
		1	SCA6 enabled
	D6-D0	0-31	SCA6 Upper Threshold MSB
56	D7-D0	0-255	SCA7 Lower Threshold LSB
57	D7-D0	0-31	SCA7 Lower Threshold MSB
58	D7-D0	0-255	SCA7 Upper Threshold LSB
59	D7	0	SCA7 disabled
		1	SCA7 enabled
	D6-D0	0-31	SCA7 Upper Threshold MSB
60	D7-D0	0-255	SCA8 Lower Threshold LSB
61	D7-D0	0-31	SCA8 Lower Threshold MSB
62	D7-D0	0-255	SCA8 Upper Threshold LSB
63	D7	0	SCA8 disabled
		1	SCA8 enabled
	D6-D0	0-31	SCA8 Upper Threshold MSB

Table 1 - PX4 Analog Gain Settings

Analog Gain	Analog Gain Control A (Config byte 8, bit D5)	Analog Gain Control B (Config byte 15, bits D3-D0)
4.13x	1	0
4.95x	1	1
5.94x	1	2
7.17x	1	3
8.22x	0	0
9.84x	0	1
11.8x	0	2
14.3x	0	3
17.5x	1	4
20.9x	1	5
25.1x	1	6
30.3x	1	7
34.7x	0	4
41.6x	0	5
49.9x	0	6
60.3x	0	7
76.7x	0	8
91.9x	0	9
110x	0	10
133x	0	11
163x	1	12
195x	1	13
234x	1	14
283x	1	15
324x	0	12
388x	0	13
466x	0	14
563x	0	15

Table 2 – PX4 Peaking Time Settings

Peaking Time (uS)	Flat Top (uS)	Decimation	'Rise' register	Flat Top register	Decimation register
0.8	0.2...3.2	1	1	0…15	0
1.6	0.2...3.2	1	2	0…15	0
2.4	0.2...3.2	1	3	0…15	0
3.2	0.2...3.2	1	4	0…15	0
4.0	0.2...3.2	1	5	0…15	0
4.8	0.2...3.2	1	6	0…15	0
5.6	0.2...3.2	1	7	0…15	0
6.4	0.2...3.2	1	8	0…15	0
8.0	0.4...6.4	2	5	0…15	1
9.6	0.4...6.4	2	6	0…15	1
11.2	0.4...6.4	2	7	0…15	1
12.8	0.4...6.4	2	8	0…15	1
16.0	0.8...12.8	4	5	0…15	2
19.2	0.8...12.8	4	6	0…15	2
22.4	0.8...12.8	4	7	0…15	2
25.6	0.8...12.8	4	8	0…15	2
32.0	1.6...25.6	8	5	0…15	3
38.4	1.6...25.6	8	6	0…15	3
44.8	1.6...25.6	8	7	0…15	3
51.2	1.6...25.6	8	8	0…15	3
64.0	3.2...51.2	16	5	0…15	4
76.8	3.2...51.2	16	6	0…15	4
89.6	3.2...51.2	16	7	0…15	4
102.4	3.2...51.2	16	8	0…15	4

3.1.3 USB Status Packet

Offset	Data bits	Allowed value	Description
0	D7-D0	0-255	Fast count LSB (Buffer A or B, depending on packet request)
1	D7-D0	0-255	Fast count byte 2 (Buffer A or B, depending on packet request))
2	D7-D0	0-255	Fast count byte 3 (Buffer A or B, depending on packet request))
3	D7-D0	0-255	Fast count MSB (Buffer A or B, depending on packet request))
4	D7-D0	0-255	Slow count LSB (Buffer A or B, depending on packet request))
5	D7-D0	0-255	Slow count byte 2 (Buffer A or B, depending on packet request))
6	D7-D0	0-255	Slow count byte 3 (Buffer A or B, depending on packet request))
7	D7-D0	0-255	Slow count MSB (Buffer A or B, depending on

			packet request))
8	D7-D4	3-15	FPGA version, major
	D3-D0	0-15	FPGA version, minor
9	D7-D0	0-99	Acc. Time (0-99, 1mS/count)
10	D7-D0	0-255	Acc. Time LSB, 100mS/count
11	D7-D0	0-255	Acc. Time byte 2
12	D7-D0	0-255	Acc. Time MSB
13	D7-D4	3-15	Firmware version, major
	D3-D0	0-15	Firmware version, minor
14	D7-D0	0-255	Serial Number LSB
15	D7-D0	0-255	Serial Number byte 2
16	D7-D0	0-255	Serial Number byte 3
17	D7-D0	0-255	Serial Number MSB
18	D7-D4	0	(unused)
	D3-D0	0-15	HV MSB
19	D7-D0	0-255	HV LSB (0.5V/count)
20	D7-D4	0	(unused)
	D3-D0	0-15	Detector temperature MSB
21	D7-D0	0-255	Detector temperature LSB (0.1 degree Kelvin/count)
22	D7-D0	-128…+127	Board temp (1 °C/count, signed)
23	D7	1	PX4 detected
	D6	0	Auto Fast Threshold not locked
		1	Auto Fast Threshold locked
	D5	0	MCA disabled
		1	MCA enabled
	D4	0	Preset count not reached
		1	Preset Count reached, MCA disabled
	D3	0	PX4 power supplies are OFF
		1	PX4 power supplies are ON
	D2	0	Oscilloscope data not ready
		1	Oscilloscope data ready
	D1	0	Unit is unconfigured
		1	Unit is configured
	D0	0	No power button configuration occurred
		1	Power button configuration occurred

24	D7-D0	0-255	General Purpose Counter LSB
25	D7-D0	0-255	G. P. Counter byte 2
26	D7-D0	0-255	G. P. Counter byte 3
27	D7-D0	0-255	G. P. Counter MSB
28	D7	0	Auto Input Offset locked
		1	Auto Input Offset searching
	D6	0	MCS not finished
		1	MCS finished
	D5-D0	N/A	
29	D7-D0	0-255	DCAL DAC setting, LSB
30	D7-D4	0	N/A
	D3-D0	0-15	DCAL DAC setting, MSB
31	D7-D0	0-255	TECCL DAC setting, LSB
32	D7-D4	0	N/A
	D3-D0	0-15	TECCL DAC setting, MSB
33-63	D7-D0	0	N/A (Currently unused)

This document reflects the following versions:

FP4.00 (FPGA)

FW4.01 (Firmware)