ProSensing Application Framework:
Network Communication Protocol

December 6, 2011

1. Document Summary

The network module component of the common application framework offers a common client-server communication protocol applicable to all implementations of the framework.

2. Server Status Codes

Name	32-bit Code	Description
NETRES_SRV_ERR	65	Server error
NETRES_OK	66	Command succeeded
NETRES_CFG_TRANSITION	67	Server configuration transition is in progress
NETRES_LACK_CONTROL	68	Client lacks control status (**Set Config** command)
NETRES_UNKNOWN_CMD	69	Unrecognized command code
NETRES_UNKNOWN_DATA_TYPE	70	Unknown data product type requested in **Get Data** command
NETRES_WRONG_DATA_SIZE	71	Incorrect configuration size from client in **Set Config** command
NETRES_NO_DATA	72	There are no data blocks of a requested type available that haven't already been delievered to the client. **NOTE:** This status code is returned even if the requested data type is inapplicable to the current processing mode, in which case no data of the requested type would be generated. (**Get Data** command)
NETRES_WRONG_ARCHIVE	73	The archive index supplied by the client does not match the archive index currently in effect on the server. In general, the client should respond to this error with a "Get Configuration" command to freshen its local copy of the server configuration.
NETRES_INVALID_INDEX	74	An index supplied by the client is outside the valid range
NETRES_INVALID_PARAM	75	A parameter supplied by the client is invalid
NETRES_INVALID_TEXT	76	Pacsi script contains errors
NETRES_NETCMD_BUSY	77	Most recently accepted net command is still busy

3. Client → Server Commands

3.a Preface: Transactions

3.a.1 Length

- Unless otherwise specified, any given transaction component of length 4 is a 32-bit, 2's complement signed integer, packed little endian.

3.a.2 Initial Response Codes

- Unless otherwise specified, the initial response code of any network command may be NETRES_SRV_ERR, an indication that the server momentarily encountered or is continually experiencing an error condition that cannot be further described to the client via the network command protocol; it is implied that said error condition prevents the server from evaluating the command. In such an event, the server log file should contain additional information.

3.1 Ping

3.1.1 Code: 0x01

3.1.2 Description

An affirmative response to a Ping request indicates that the remote server application is running. The simplest of all command transactions, the server's response consists only of NETRES_OK.

3.1.3 Transaction

Ping			
Direction	**Type**	**Length (bytes)**	**Value**
client ⇨ server	Request code	4	0x01
server ⇨ client	Final status code	4	NETRES_OK

3.2 Get Configuration

3.2.1 Code: 0x02

3.2.2 Description

Request the current server configuration. The returned configuration structure is application-specific.

3.2.3 Transaction

Get Configuration (with optionally attached status)			
Direction	**Description**	**Length (bytes)**	**Value**
client ⇨ server	Request code	4	0x02: Configuration only 0x08: Configuration with status
server ⇨ client	Initial response code	4	NETRES_OK on success, NETRES_CFG_TRANSITION if the server configuration is changing, i.e., no configuration is presently in effect
server ⇨ client	Total response size	4	12 + project-specific configuration size
server ⇨ client	Archive index	4	Unique index associated with the current server configuration. Incremented when the server configuration changes.
server ⇨ client	Project configuration size	4	Size of project-specific configuration
server ⇨ client	Project status size	4	Request code 0x02: Zero. Request code 0x08: Size of project-specific status structure.
server ⇨ client	Project configuration	project-specific size	project-specific content
server ⇨ client	Project status	Request code 0x02: 0 Request code 0x08: project-specific size	project-specific content
server ⇨ client	Final status code	4	NETRES_OK

3.3 Set Configuration

3.3.1 Code: 0x03

3.3.2 Description

Apply a new operating configuration on the server. The format of the configuration structure is application-specific.

3.3.3 Transaction

Set Configuration			
Direction	**Description**	**Length (bytes)**	**Value**
client ⇨ server	Request code	4	0x03
server ⇨ client	Initial response code	4	NETRES_OK on success, NETRES_CFG_TRANSITION if the server configuration is already changing, NETRES_LACK_CONTROL if another client already has control status
client ⇨ server	Configuration size	4	(project specific)
client ⇨ server	New configuration	(configuration size)	(project specific)
server ⇨ client	Final status code	4	NETRES_OK on success, NETRES_WRONG_DATA_SIZE on incorrect configuration size from client, NETRES_SRV_ERR if the server fails to propagate the new configuration

3.4 Get Status

3.4.1 Code: 0x04

3.4.2 Description

Request the current server status. The returned status structure is application-specific.

3.4.3 Transaction

Get Status			
Direction	**Description**	**Length (bytes)**	**Value**
client ⇨ server	Request code	4	0x04
server ⇨ client	Initial response code	4	NETRES_OK
server ⇨ client	Total response size	4	4 + status size
server ⇨ client	Status size	4	(project specific)
server ⇨ client	Status data	(status size)	(project specific)
server ⇨ client	Final status code	4	NETRES_OK

3.5 Get Server Info

3.5.1 Code: 0x05

3.5.2 Description

Request a data block containing a variety of parameters describing the server.

3.5.3 Transaction

Get Server Info			
Direction	**Description**	**Length (bytes)**	**Value**
client ⇨ server	Request code	4	0x05
server ⇨ client	Initial response code	4	NETRES_OK
server ⇨ client	ServerInfo structure size	4	380 + 128 * product count
server ⇨ client	ServerInfo structure	ServerInfo struct size	See ServerInfo structure documentation
server ⇨ client	Final status code	4	NETRES_OK

3.6 Get Data

3.6.1 Code: 0x07

3.6.2 Description

Retrieve a given number of data blocks of a given product type from the server.

3.6.3 Transaction

Get Data			
Direction	**Description**	**Length (bytes)**	**Value**
client ⇨ server	Request code	4	0x07
client ⇨ server	ClientDataRequest structure size	4	60
client ⇨ server	ClientDataRequest structure	60	Information identifying the type of product and desired format of the returned data
server ⇨ client	Initial response code	4	NETRES_OK on success, NETRES_WRONG_DATA_SIZE on incorrect ClientDataRequest structure size, NETRES_CFG_TRANSITION if a configuration change is in progress, NETRES_UNKNOWN_DATA_TYPE if the requested data product type code is not recognized, NETRES_NO_DATA if no new data is available for the requested product type (*note that this code is returned* ***even if no data of the requested type is generated in the current processing mode***), NETRES_WRONG_ARCHIVE if the expected archive index specified in the ClientDataRequest structure is not equal to -1 and does not match the archive index of the server configuration currently in effect, NETRES_INVALID_INDEX if any matrix subset coordinate is outside the product data matrix
server ⇨ client	ServerDataResponse structure size	4	272 + total data size (# blocks * block size)
server ⇨ client	ServerDataResponse structure	272 + total data size (# blocks * block size)	Actual returned data with a descriptive header structure attached
server ⇨ client	Final status code	4	NETRES_OK

3.7 Load PACSI Script

3.7.1 Code: 0x200 or 0x201

3.7.2 Description

Load a given Pacsi script on the server, and immediately begin executing it if no problems are discovered, optionally in parallel with any already running script(s). If there are any problems with the script, the server communicates a textual error summary to the client.

3.7.3 Transaction

Load PACSI Script			
Direction	**Description**	**Length (bytes)**	**Value**
client ⇨ server	Request code	4	0x200: Pedestal commands disabled 0x201: Pedestal commands enabled
server ⇨ client	Initial response code	4	NETRES_OK
client ⇨ server	Client data size	4	Total size, in bytes, of client data, which is the Pacsi script size + 1 (the ”Run in parallel” boolean parameter is 1 byte)
client ⇨ server	Run in parallel	1	Boolean value; 0: If any other Pacsi script is running on the server, it is stopped, and this one replaces it. 1: Any Pacsi script already running on the server continues untouched; the new script is started in parallel.
client ⇨ server	Pacsi script text	Pacsi script text size	The ASCII text comprising the Pacsi script to be loaded
server ⇨ client	Compilation result code	4	Result of attempted compilation of given Pacsi script: NETRES_SRV_ERR if either the initial pedestal STOP command fails, or there is an internal problem with an already-running Pacsi script, NETRES_INVALID_TEXT if there is at least one problem with the given Pacsi script; see next transaction component
server ⇨ client	Number of problems with Pacsi script	4	An integer that is the number of errors encountered while compiling the given Pacsi script

server ⇨ client	Size of error explanation text	4	Size, in bytes, of the following ASCII text describing the Pacsi script error(s); if the number of errors is zero, this too is zero
server ⇨ client	Explanation of Pacsi error(s)	Size of error text; see preceding transaction component	ASCII text describing the problem(s) with the given Pacsi script; if there are no errors, then this transaction component is omitted

Appendix

A. Framework Data Structures

A.a Preface: Data Types

A.a.1 Integers

- Unless otherwise specified, any given "integer" component of size 4 is a 32-bit, 2's complement signed integer, packed little endian.

A.1 PedVector structure

Size: 48 bytes

Parameter	Data type	Size (bytes)	Description
Position, azimuth	double	8	Azimuth position in degrees
Position, elevation	double	8	Elevation position in degrees
Velocity, azimuth	double	8	Azimuth-axis velocity in degrees
Velocity, elevation	double	8	Elevation-axis velocity in degrees
Motor current, azimuth	double	8	Azimuth-axis motor current in amps
Motor current, elevation	double	8	Elevation-axis motor current in amps

A.2 GeoVector structure

Size: 52 bytes

Parameter	Data type	Size (bytes)	Description
Latitudinal hemisphere	integer	4	0: Undefined 'N' (ascii 78): North 'S' (ascii 83): South
Position, latitude	double	8	Latitudinal position in degrees
Longitudinal hemisphere	integer	4	0: Undefined 'E' (ascii 69): East 'W' (ascii 87): West
Position, longitude	double	8	Longitudinal position in degrees
Altitude	double	8	Altitude in meters
Heading reference	integer	4	0: Undefined 'T' (ascii 84): True north 'M' (ascii 77): Magnetic north

Heading	double	8	Heading in degrees
Speed	double	8	Speed in km/hour

A.3 ServerInfo structure

Size: 380 + 128 * <product count> bytes

Parameter	Data type	Size (bytes)	Description
Project name	text string	128	The name of the project associated with the server. Unused bytes are set to 0.
Digital receiver info	DigRcvInfo structure	248	Information describing the digital receiver model in use by the server application.
Product count	integer	4	# of data product types available on the server via the Get Data command.
Data product info	array of DataProductInfo structures	128 * product count	One DataProductInfo structure for each data product type available on the server.

A.4 DigRcvInfo structure

Size: 248 bytes

Parameter	Data type	Size (bytes)	Description
Manufacturer code	integer	4	Digital receiver manufacturer identification 1: Offline (simulated receiver) 2: Echotek (Mercury Computer Systems)
Manufacturer name	text string	32	Digital receiver manufacturer name
Model code	integer	4	Digital receiver model identification 1: Offline (simulated receiver) 2: Echotek ECDR-2-12210 3: Echotek ECV4-2R105 4: Echotek ECV4-2R130-SL
Model name	text string	32	Digital receiver model name
Specification info	DigRcvSpec structure	176	Receiver specification

A.5 DataProductInfo structure

Size: 128 bytes

Parameter	Data type	Size (bytes)	Description
Type code	integer	4	Unique integer associated with product type
Type name, short	text string	32	Condensed name of product type
Type name, long	text string	64	Descriptive name of product type
Digital receiver channel	integer	4	The zero-based index of the digital receiver channel that was the source of the product data, or -1 if not applicable. A data product is considered to be associated with a digital receiver channel if the data has not been processed by a custom, application-supplied algorithm, i.e., "raw" samples from the digital receiver with any amount of built-in (framework-supplied) pre-processing applied meet this condition. The channel index is -1 for all other data products, i.e., those not generated automatically by the framework.
Positioner device index	integer	4	Unique integral identifier of positioner associated with the data product, or -1 if N/A
GPS device index	integer	4	Unique integral identifier of GPS associated with the data product, or -1 if N/A
Data domain type	integer	4	Mathematical domain of data matrix 0: Undefined 1: Time domain 2: Frequency domain
Data unit type	integer	4	Unit of measurement of each data value 0: Undefined 1: Digital counts 2: Digital counts ^{2} 3: Voltage 4: Voltage ^{2} 5: mWatt 6: dBm 7: dBZ

A.5 *(continued)* DataProductInfo structure

Parameter	Data type	Size (bytes)	Description
# interleaved tracks	integer	4	# of interleaved data tracks per vector; typically this is either '1' (no interleaving) for real-valued data, or '2' for complex-valued data, e.g., in-phase/quadrature pairs
# matrix dimensions	integer	4	# of dimensions spanned by a complete data product block; a single real or complex value is '0', a vector of values is '1', a collection of fixed-length vectors is '2', and so on; the framework supports up to 4 dimensions of data

A.6 DigRcvSpec structure

Size: 176 bytes

Parameter	Data type	Size (bytes)	Description
# input channels	integer	4	Number of analog input channels
Min sample clock frequency, Hz	integer	4	Minimum sample clock frequency
Max sample clock frequency, Hz	integer	4	Maximum sample clock frequency
Min center frequency, Hz	integer	4	Minimum center frequency (on-board NCO)
Max center frequency, Hz	integer	4	Maximum center frequency (on-board NCO)
Min # DMA descriptors	integer	4	Minimum # DMA descriptors
Max # DMA descriptors	integer	4	Maximum # DMA descriptors
Min # DMA descriptor packets	integer	4	Minimum # packets per DMA descriptor
Max # DMA descriptor packets	integer	4	Maximum # packets per DMA descriptor
Min DMA packet size, bytes	integer	4	Minimum DMA packet size
Max DMA packet size, bytes	integer	4	Maximum DMA packet size
DMA packet size granularity, bytes	integer	4	Factor that must evenly divide the DMA packet size
Min burst size, bytes	integer	4	Minimum size of each triggered data burst
Max burst size, bytes	integer	4	Maximum size of each triggered data burst
Burst size granularity, bytes	integer	4	Factor that must evenly divide the burst size
Min # bursts per PCI interrupt	double	8	Minimum # triggered data bursts per PCI interrupt (one interrupt per DMA collection)
Max # bursts per PCI interrupt	double	8	Maximum # triggered data bursts per PCI interrupt (one interrupt per DMA collection)
Min skip count, samples	integer	4	Minimum offset, in units of decimated samples, from the data trigger to the first collected sample in the data burst. This should always be zero.
Max skip count, samples	integer	4	Maximum offset, in units of decimated samples, from the data trigger to the first collected sample in the data burst.
Min CIC decimation level	integer	4	Minimum decimation level for CIC filter
Max CIC decimation level	integer	4	Maximum decimation level for CIC filter (1 if the feature is not available)

A.6 *(continued)* DigRcvSpec structure

Parameter	Data type	Size (bytes)	Description
Min FIR decimation level	integer	4	Minimum decimation level following application of FIR filter
Max FIR decimation level	integer	4	Maximum decimation level following application of FIR filter (1 if the feature is not available)
Min post decimation level	integer	4	Minimum final on-board decimation level
Max post decimation level	integer	4	Maximum final on-board decimation level (1 if the feature is not available)
Max FIR filter length, # weights	integer	4	Maximum length of FIR filter
Min FIR gain	double	8	Minimum FIR filter gain level, measured as the area of the FIR filter, in counts^2
Max FIR gain	double	8	Maximum FIR filter gain level, measured as the area of the FIR filter, in counts^2
Full-scale FIR gain level	double	8	Full-scale (unity) FIR filter gain level, measured as the area of the FIR filter, in counts^2
Min analog voltage input, V	double	8	Minimum analog voltage that will not saturate the A/D converter (generally = -1 * max analog voltage)
Max analog voltage input, V	double	8	Maximum analog voltage that will not saturate the A/D converter
Analog impedance, ohms	double	8	Impedance of each analog input channel
Min digitized count value	integer	4	Minimum value of a sample delivered to the application
Max digitized count value	integer	4	Maximum value of a sample delivered to the application
A/D resolution, bits/sample	integer	4	Resolution of the A/D converter
Digitized count size, bits	integer	4	Size of each digital count actually delivered to the application; the samples are scaled such that the maximum count at the A/D converter corresponds to a delivered count of $2^{\text{countSizeBits}} - 2^{(\text{countSizeBits} - \text{ADResolutionBits})}$

A.7 ClientDataRequest structure

Size: 60 bytes

Parameter	Data type	Size (bytes)	Description
Product type code	integer	4	Integer associated with desired product type. A list of available product types and their respective codes can be retrieved via the Get Server Info command.
Block offset	integer	4	Index of first block to be returned to the client, relative to the newest group of product data blocks. A value of -1 selects the oldest data block that has not yet been delivered to the client issuing the request. For the vast majority of applications, this should always be -1.
Block step length	integer	4	When returning multiple blocks (i.e. max # blocks > 1 and more than one "unseen" block is available), the indices of any two sequential returned blocks differ by this step length. For the vast majority of applications, this should always be 1.
Block seen step length	integer	4	When the server observes that a particular block in its data stream has already been delivered to the client in response to a past request, the server "skips ahead" by this many blocks until it either lands on a new block (i.e., unseen by the client), or arrives at the end of the data stream. For the vast majority of applications, this should always be 1.
Max # blocks	integer	4	Desired quantity of data blocks to be sent to the client. Fewer than this many blocks may actually be sent, depending on availability. Note that a value of zero is perfectly valid; the server will then return only the data header information, i.e., a ServerDataResponse structure.

A.7 *(continued)* ClientDataRequest structure

Matrix subset selection: extent 1	Array [4] of 32-bit integers	4 * 4 = 16	The matrix selection extents are the coordinates of the "corners" defining the span of the desired matrix region. A coordinate of -1 (2's complement) selects the maximum index of its respective dimension. For example, if the parent matrix (on the server) for a particular product type is of dimensions N x M, and we wish to select all of row R, then extent 1 = {0, 3, *, *}, and extent 2 = {-1, 3, *, *}, where * is any integer (the 3rd and 4th dimension coordinates are not applicable to a two dimensional N x M matrix).
Matrix subset selection: extent 2	Array [4] of 32-bit integers	4 * 4 = 16	
Expected archive index	integer	4	Archive index expected to be associated with the returned data. A unique archive index is assigned to each server configuration, so this parameter may be used to ensure that consistency is maintained between the client-side copy of the current configuration and the transmitted data blocks. A value of -1 (2's complement) bypasses this feature; any available data block of the given type will be sent.
Format flags	integer	4	Bitflags that modify the default format of the returned data: 0x01: Return data points as single-precision floating-point (32-bit float) 0x02: Return data points as double-precision floating-point (64-bit double); **this is the default data format** 0x04: Return data points as 16-bit signed integers (2's complement) 0x08: Return data points as 32-bit signed integers (2's complement) 0x10: Return data points as 64-bit signed integers (2's complement) 0x10000: Return **oldest** data block that has not yet been "seen" by (i.e., delivered to) the client; the default behavior is to always return the **newest** data block available, if it hasn't yet been "seen".

A.8 ServerDataResponse structure

Size: 272 bytes + data size

Parameter	Data type	Size (bytes)	Description
Data product type code	integer	4	Unique integral identifier associated with the desired data product type (project specific)
Time stamp, seconds	integer	4	Time at which the raw receiver data associated with the product data was collected (seconds)
Time stamp, microseconds	integer	4	Time at which the raw receiver data associated with the product data was collected (*u*seconds)
Digital receiver configuration	DigRcvConfig structure	108	The configuration of the digital receiver that gathered the data
Digital receiver FIR filter gain	double	8	Gain factor by which the data from the digital receiver was scaled following time-domain convolution with the FIR filter
Averaging interval length	integer	4	Length of the averaging interval associated with the data, or '1' if there is no such association.
Archive index	integer	4	Unique index associated with the current server configuration. Incremented when the server configuration changes.
Block index	integer	8 (64-bit int)	Unique index associated with the first returned data block. Reset to zero when the server configuration changes, i.e., this index is unique only in the context of the current archive index.
# blocks	integer	4	# of data product blocks returned
Block size	integer	4	Size, in bytes, of each product data block
# block dimensions	integer	4	# of dimensions spanned by each product data block. For most product types, this is = 1.

A.8 *(continued)* ServerDataResponse structure

Parameter	Data type	Size (bytes)	Description
Block dimension sizes	Array [4] of 32-bit integers	4 * 4 = 16	For 1-dimensional data products, i.e., 1 x N matrix, array element [0] is simply equal to the block size in bytes, so N = array[0] / 8. For 2-dimensional data, i.e., N x M matrix, N = array[0] / 8, and M = array[1]. Note that only array[0] is in units of bytes.
Pedestal status	PedVector structure	48	Pedestal status as recorded at the collection time of the data, if applicable
GPS status	GeoVector structure	52	GPS status as recorded at the collection time of the data, if applicable
Product data	Array length = # blocks * blockSize / datumSize Array element type = (dependent on "Format flags" field of ClientDataRequest)	# blocks * block size	Actual product data. Each value is a single double-precision floating-point for real-valued data, and a real/imaginary double-precision floating-point pair for complex-valued data, unless: **NOTE:** The "Format flags" field of the ClientDataRequest structure may be used to request a numeric format other than double-precision; please see ClientDataRequest::formatFlags.

A.9 DigRcvConfig structure

Size: 108 bytes

Parameter	Data type	Size (bytes)	Description
Mode	integer	4	0 = Burst; 1 = Gated; 2 = PRI
Trigger source	integer	4	0 = Internal 1 = External (e.g., RCB-triggered)
PCI bus frequency, MHz	integer	4	PCI bus frequency (33, 66, or 133)
A/D sample clock frequency, Hz	integer	4	The frequency at which samples are collected
# DMA descriptors	integer	4	# of DMA descriptors allocated to the digital receiver
# DMA packets per descriptor	integer	4	# of DMA packets within each descriptor
DMA packet size	integer	4	Size, in bytes, of each DMA packet
Block size	integer	4	Size, in bytes, of each triggered data block collected by the receiver
Reserved	integer	4	Reserved for internal use - ignore
# blocks per PCI interrupt	double	8	# triggered data bursts per PCI interrupt (one interrupt per DMA collection)
Reserved	integer	4	Reserved for internal use - ignore
Reserved	integer	4	Reserved for internal use - ignore
State (one value per channel)	integer [2]	4 * 2 = 8	Collection state of each receiver channel. 0 = Idle 1 = Run
Data source (one value per channel)	integer [2]	4 * 2 = 8	Source from which data is collected. 0 = Raw A/D samples 1 = Ramp (test counter) 2 = Digital tuner
Skip count (one value per channel)	integer [2]	4 * 2 = 8	# of samples that elapse between the data trigger and the first collected sample
CIC decimation (one value per channel)	integer [2]	4 * 2 = 8	CIC filter decimation; precedes FIR filter. Not supported on all receivers.
FIR decimation (one value per channel)	integer [2]	4 * 2 = 8	FIR filter decimation; follows CIC filter
Post decimation (one value per channel)	integer [2]	4 * 2 = 8	Final decimation layer. Not supported on all receivers.
NCO frequency, Hz (one value per channel)	integer [2]	4 * 2 = 8	Frequency of the receiver's onboard numerically controlled oscillator, which acts as the center frequency for I/Q detection.

B. X-Pol Specific Data Structures

B.1 X-Pol configuration structure

Size: 1324 bytes

Parameter	Data type	Size (bytes)	Description
Radar site info: text description	text string	1024	*Please refer to the document* "Pacsi_SetConfig_params_xpol" *for a detailed description of each XPol configuration parameter*
Radar site info: azimuth offset	double	8	
(spare)	-	4	
Clutter filter width, m/sec	double	8	
Clutter averaging interval	integer	4	
Data rate, # products/sec	double	8	
FFT length	integer	4	
FFT window type	integer	4	
(reserved)	integer	4	
Auto file-roll: # records	integer	4	
Auto file-roll: # scans	integer	4	
Auto file-roll: file size, MB	integer	4	
Auto file-roll: elapsed time, sec	integer	4	
Auto file-roll type, bitflags	integer	4	
(reserved)	integer	4	
Filter bandwidth, MHz	double	8	
Frequency tracking adjust threshold, %	double	8	
Frequency tracking mode	integer	4	
(reserved)	integer	4	
Group interval, usec	integer	4	
'H' dBZ / dBm offset	double	8	
'H' noise power, dBm	double	8	
Integration time, sec	double	8	
LO frequency error, MHz	double	8	
LO frequency, MHz	double	8	
Max sampled range, m	double	8	

B.1 *(continued)* X-Pol configuration structure

Parameter	Data type	Size (bytes)	Description
# range gates	integer	4	
# group pulses	integer	4	
Post decimation level	integer	4	
Post averaging interval	integer	4	
PRI, usec: unit 1	integer	4	
PRI, usec: unit 2	integer	4	
PRI, usec: total	integer	4	
Primary on-board decimation level (CIC filter)	integer	4	
Pulse length, m	double	8	
(reserved)	integer	4	
Range gate spacing, m	double	8	
(reserved)	integer	4	
Range resolution, m / gate	double	8	
Record moments (true/false)	integer	4	
Record raw (true/false)	integer	4	
Recording enabled (true/false)	integer	4	
Server mode	integer	4	
(reserved)	integer	4	
Server state	integer	4	
(reserved)	integer	4	
Software decimation level	integer	4	
Sum powers (true/false)	integer	4	
Total averaging interval	integer	4	
Tx delay, nsec	integer	4	
Tx delay, pulse width multiplier	integer	4	
Tx pulse center, nsec	integer	4	
Tx pulse center offset, nsec	integer	4	
Tx sample switch delay, nsec	integer	4	
Tx sample switch holdoff, nsec	integer	4	

B.1 *(continued)* X-Pol configuration structure

Parameter	Data type	Size (bytes)	Description
Use clutter filter (true/false)	integer	4	
'V' dBZ / dBm offset	double	8	
'V' noise power, dBm	double	8	
Zero range gate index	double	8	

B.2 X-Pol status structure

Size: 68 bytes

Parameter	Data type	Size (bytes)	Description
Time stamp, seconds	integer	4	Time at which status was valid
Time stamp, microseconds	integer	4	Time at which status was valid
Radar temperatures	integer [4]	4 * 4 = 16	Raw RTD values from RCB
Inclinometer, roll	integer	4	Raw voltage from RCB
Inclinometer, fore/aft	integer	4	Raw voltage from RCB
Fuel sensor	integer	4	Raw value from RCB; ignore if N/A
CPU temperature	float	4	Temperature of CPU chip in Celsius
Pedestal scan type	integer	4	Unique integer associated with current pedestal scan type 0: None 1: Custom scan pattern 2: Soft stop (slowing to a halt) 3: Point 4: Slew 5: Azimuth scan (PPI) 6: Elevation scan (RHI) 7: Azimuth raster 8: Elevation raster 9: Volume 10: (paused)
Tx power, mW	float	4	Measured radar transmit power
Reserved	integer [5]	7 * 4 = 28	Spares

C. X-Pol Specific Numeric Enumerations

C.1 X-Pol data types

The following data types are defined for the XPol project, and are available in real-time via the **GetData** network command.

Name	Type ID	Matrix dimensions (1 x N vectors omit "1 x")	Data domain
Digital receiver filtered I/Q, V chan	0	# gates X software decimation level	Complex
Digital receiver filtered I/Q, V chan, with phase alignment and clutter filtering	1	# gates X software decimation level	Complex
Digital receiver filtered I/Q, H chan	2	# gates X software decimation level	Complex
Digital receiver filtered I/Q, H chan, with phase alignment and clutter filtering	3	# gates X software decimation level	Complex
Received power, V0, unaveraged	4	# gates	Real
Received power, V1, unaveraged	5	# gates	Real
Received power, V2, unaveraged	6	# gates	Real
Received power, H0, unaveraged	7	# gates	Real
Received power, H1, unaveraged	8	# gates	Real
Received power, H2, unaveraged	9	# gates	Real
Total summed power, V, unaveraged	10	# gates	Real
Total summed power, H, unaveraged	11	# gates	Real
Doppler pulse pair, V0, V1, unaveraged	12	# gates	Complex
Doppler pulse pair, V1, V2, unaveraged	13	# gates	Complex
Doppler pulse pair, H0, H1, unaveraged	14	# gates	Complex
Doppler pulse pair, H1, H2, unaveraged	15	# gates	Complex
Cross correlation, V, H, unaveraged	16	# gates	Complex
Power spectra, V0, unaveraged	17	# gates X FFT length	Real
Power spectra, V1, unaveraged	18	# gates X FFT length	Real
Power spectra, H0, unaveraged	19	# gates X FFT length	Real
Power spectra, H1, unaveraged	20	# gates X FFT length	Real
Cross spectra, V0, H0, unaveraged	21	# gates X FFT length	Complex
Cross spectra, V1, H1, unaveraged	22	# gates X FFT length	Complex

C.1 *(continued)* X-Pol data types

Name	Type ID	Matrix dimensions (1 x N vectors omit "1 x")	Data domain
Received power, V0, averaged	23	# gates	Real
Received power, V1, averaged	24	# gates	Real
Received power, V2, averaged	25	# gates	Real
Received power, H0, averaged	26	# gates	Real
Received power, H1, averaged	27	# gates	Real
Received power, H2, averaged	28	# gates	Real
Total summed power, V, averaged	29	# gates	Real
Total summed power, H, averaged	30	# gates	Real
Doppler pulse pair, V0, V1, averaged	31	# gates	Complex
Doppler pulse pair, V1, V2, averaged	32	# gates	Complex
Doppler pulse pair, H0, H1, averaged	33	# gates	Complex
Doppler pulse pair, H1, H2, averaged	34	# gates	Complex
Cross correlation, V, H, averaged	35	# gates	Complex
Power spectra, V0, averaged	36	# gates X FFT length	Real
Power spectra, V1, averaged	37	# gates X FFT length	Real
Power spectra, H0, averaged	38	# gates X FFT length	Real
Power spectra, H1, averaged	39	# gates X FFT length	Real
Cross spectra, V0, H0, averaged	40	# gates X FFT length	Complex
Cross spectra, V1, H1, averaged	41	# gates X FFT length	Complex
IF phasor estimate (frequency tracking)	42	1	Complex

C.1 *(continued)* X-Pol data types

<table>
<tr><th>Name</th><th>Type ID</th><th>Matrix dimensions
(1 x N vectors omit “1 x”)</th><th>Data domain</th></tr>
<tr><td>Fused products, digital receiver data</td><td>43</td><td>Product 1 (V-chan): # gates
Product 2 (H-chan): # gates</td><td>Complex</td></tr>
<tr><td>Fused products, processed data</td><td>44</td><td colspan="2">
<table>
<tr><th>Mode</th><th>Products</th></tr>
<tr><td>SPP</td><td>Recv power, V0, averaged
Recv power, V1, averaged
Recv power, H0, averaged
Recv power, H1, averaged
Doppler PP, V0, V1, averaged
Doppler PP, H0, H1, averaged
Cross corr, V, H, averaged</td></tr>
<tr><td>SPP, sum powers</td><td>Total power, V, averaged
Total power, H, averaged
Doppler PP, V0, V1, averaged
Doppler PP, H0, H1, averaged
Cross corr, V, H, averaged</td></tr>
<tr><td>DPP</td><td>Recv power, V0, averaged
Recv power, V1, averaged
Recv power, V2, averaged
Recv power, H0, averaged
Recv power, H1, averaged
Recv power, H2, averaged
Doppler PP, V0, V1, averaged
Doppler PP, V1, V2, averaged
Doppler PP, H0, H1, averaged
Doppler PP, H1, H2, averaged
Cross corr, V, H, averaged</td></tr>
<tr><td>DPP, sum powers</td><td>Total power, V, averaged
Total power, H, averaged
Doppler PP, V0, V1, averaged
Doppler PP, V1, V2, averaged
Doppler PP, H0, H1, averaged
Doppler PP, H1, H2, averaged
Cross corr, V, H, averaged</td></tr>
<tr><td>FFT</td><td>Power spectra, V0, averaged
Power spectra, H0, averaged
Cross spectra, V0, H0, averaged</td></tr>
<tr><td>FFT-Alt,
FFT-Int</td><td>Power spectra, V0, averaged
Power spectra, V1, averaged
Power spectra, H0, averaged
Power spectra, H1, averaged
Cross spectra, V0, H0, averaged
Cross spectra, V1, H1, averaged</td></tr>
</table>
</td></tr>
</table>