Plasma Calibration System: Network Specification

Overview

Data Types

All multi-byte values are sent in network byte order (big endian). Integer values are signed and encoded in two's complement format. Floating point values are encoded according to the IEEE 754-2008 standard, and may include special values such as Infinity or Not-a-Number (Na N). String values may include Unicode data and are encoded using UTF-8. Strings are always variable length and include their length in bytes at the beginning of the string.

Name	Length (bytes)	Description
int32	4	32-bit two's complement integer
float64	8	64-bit IEEE floating point number
utf8	4 + length	32-bit integer indicating string length, followed by UTF-8 Unicode string
binary	4 + length	32-bit integer indicating data length, followed by raw binary data

Packet Structure

All packets sent by all parts of the PCS system have the same packet header. The last value in the header is the length, in bytes, of the rest of the data in the packet. Therefore, if new packet types are introduced, a component of the system can choose to ingore any packets that it does not know how to handle.

Offset	Name	Data Type	Description
0	Sync	int32	Sync Code (0xDEADBEEF)
4	Packet Type	int32	Type Code
8	Packet ID	int32	Unique Packet Identifier
12	Session Time	float64	Session Time Stamp
20	Packet Time	float64	Packet Time Stamp
28	Length	int32	Data Length (in bytes)
32	Data	varies	Data specific to packet type

Specific Values

Name	Data Type	Contents
Packet ID	int32	A sequentially incrementing number used to match up packets with their acknowledgements
Packet Time	float64	The number of seconds past January 1, 1970, 00:00:00 GMT
Property ID	int32	See section on Hardware Control Properties below

Packet Types

Type	Name	Description	Data Flow	Acknowledged?
0	Acknowledge	Confirms receipt of packet and success state	GSE ← GUI, GUI ← HW	-
1	Configuration	Complete XML Configuration of a single hardware control program	GUI ← HW	-
2	Monitor	Latest changed set and readback values from hardware properties	GSE ← GUI ← HW	-
3	Metadata	Metadata about hardware properties, such as name, error ranges, and calibration values	GSE ← GUI ← HW	-
4	Message	General string messages	GSE ← GUI, GUI ← HW	-
5	Command	Low-level control of hardware controllers	GUI → HW	yes
6	Script	High-level control of system through Java Script commands	GSE → GUI	yes
7	Capture	Binary data from "capture" hardware, such as images or spectrograms	GUI ← HW	-
8	Control	Request external control of the system	GSE → GUI	yes
9	Disconnect	Expected disconnect notification	GSE ← GUI, GUI ← HW	-
10	Identity	Announce identity of connecting software	GSE → GUI	yes
11	Event	Notify of special hardware events	GUI ← HW	-
12	Completion	Notify success or failure of script execution	GSE ← HW	-

Acknowledge Packet

An Acknowledge Packet is returned after receipt of a Command or Script packet in order to indiciate that the specified command was received and able to be processed. If the command parameters are invalid or the script is unparsable, an error response code will be returned in the Acknowledge Packet, along with an optional human-readable error message.

Name	Data Type	Description
Original ID	int32	The Unique Packet Identifier for which this is a response
Response Code	int32	A response code, 0 if successful, nonzero if there is an error
Message	utf8	A human-readable error message

Configuration Packet

Name	Data Type	Description
Configuration	binary	An XML file defining the complete initial hardware configuration

Montior Packet

A monitor packet may be empty, in which case there will only be a value indicating the hardware mode (Safe, Simulation, Operational), and the length value in the header will be 4. The length will always be a multiple of 12 bytes plus 4.

Name	Data Type	Description
Hardware Mode	int32	0 = Safe, 1 = Simulation, 2 = Operational
Property ID # 0	int32	The Unique Property Identifier indicating which value has changed
Property Value # 0	float64	The new value of this property
Property ID # 1	int32	The Unique Property Identifier indicating which value has changed
Property Value # 1	float64	The new value of this property
...		
Property ID # *n*	int32	The Unique Property Identifier indicating which value has changed
Property Value # *n*	float64	The new value of this property

Metadata Packet

More metadata may be added in future versions of this packet, but they will always be added to the end of the packet and therefore can be ignored by programs that do not know about them. Future data may include group names, display units, or other information in the configuration file if it is deemed necessary.

Name	Data Type	Description
Property ID	int32	The Unique Property Identifier indicating which metadata has changed
Property Name	utf8	The name used to identify the property in the scripting environment
Property Minimum	float64	The absolute minimum value for this property
Property Maximum	float64	The absolute maximum value for this property
Property Error Low	float64	The lower-bound value for automatic error checking and tripping (red zone)
Property Error High	float64	The upper-bound value for automatic error checking and tripping (red zone)
Property Warning Low	float64	The lower-bound value for automatic warning notification (orange zone)
Property Warning High	float64	The upper-bound value for automatic warning notification (orange zone)
Property Default Value	float64	The initial value for this property when operations or simulation mode is started
Property Safe Value	float64	The value to set this property to before safe mode is started
Property Gain	float64	An calibration gain scalar factor applied to this value
Property Offset	float64	An calibration offset applied to this value

Message Packet

Message packets are log messages passed from various programs to make it easier to understand execution flow. The messages themselves have no guarantees to structure or format, and should simply be used for logging.

Name	Data Type	Description
Log Level	int32	0=Output, 1=Warning, 2=Error, 3=Debug, 4=Journal
Log Message	utf8	The log message

Command Packet

Command packets are used to send commands to the hardware control programs in a concise manner with minimal parsing overhead. Acknowledge packets are returned to indicate success or failure of a specific command.

Name	Data Type	Description
Command Op Code	int32	The command's op code.
Command Arguments	float64...	Zero or more arguments, as determined by the op code.

Script Packet

Script packets are used to send commands to the main GUI program, and contain the same Java Script commands that can either be interactively executed or executed via a saved script. Acknowledge packets are returned to indicate success or failure in parsing and queuing execution of the script command.

Name	Data Type	Description
Script Code	utf8	The Java Script command. It can either be a single command, such as the command to launch a stored script, or an entire script in its own right.

Capture Packet

Capture packets allow hardware devices to pass large amounts of data, such as captured images or large datasets, in a single packet. The format of the capture packet depends on the hardware device in question. For example, image capture devices may send JPEG or PNG images.

Name	Data Type	Description
Property ID	int32	The Unique Property Identifier that this capture data belongs to
Capture Type	utf8	A string type identifier to indicate the data type of the capture data
Capture Data	binary	The captured data itself

Control Packet

Control packets are sent by the GSE when it wants to take external control of the system. If the GSE is not currently in control, the user will be asked to confirm whether or not they want to relinquish control, and an acknowledge packet is returned. If the user accepts, the packet's result code will indicate success, and the GSE may begin sending script packets. Any script packet sent while not in control will be ignored and an acknowledge packet indicating an error will be returned.

Name	Data Type	Description
Control Message	utf8	A message describing what task the GSE will be trying to perform to display to the user

Disconnect Packet

This packet is sent to clients so that they can detect normal disconnects. In the absence of a Disconnect Packet before a connection is terminated, it can be assumed that there was some sort of network error. Every effort will be made to send a Disconnect Packet under normal circumstances, especially if the end user forcibly closes an external connection.

Name	Data Type	Description
Disconnect Reason	int32	The reason for the expected disconnect (specific values to be provided)
Disconnect Message	utf8	A human-readable message

Identity Packet

This packet is sent by the GSE and other external connections so that they may identify themselves. (The following is not implemented but will be in the near future) Until this packet is received, no other packets will be sent.

Name	Data Type	Description
Machine ID	int32	The identity of the connecting machine/software (see table below)

Identity Assignments

Machine ID	Assignment
0	Main GUI
1	Low Voltage Hardware Controller
2	High Voltage Hardware Controller
14	GSE
15	Data Archive

Event Packet

This packet is sent by the hardware controller when an event is triggered. The GUI may respond by automatically running script and executing a script to handle the event. If an interrupting event is requested, currently executing scripts should first be aborted.

Name	Data Type	Description
Interrupt	int32	0=Non-Interrupting, Else=Interrupting
Event	utf8	A name given to the event, used for determining which event script to run

Completion Packet

This packet is sent in response to a Script packet. While the acknowledge packet is sent immediately when the script is queued for execution, this packet does not get sent until the sent script completes execution.

Name	Data Type	Description
Code	int32	0=Success, 1=Aborted, 2=Exception
Message	utf8	Error message in case of Exception, otherwise "Success" or "Aborted"

Hardware Control Properties

Each Property ID is derived from three components: the system, control, and individual property. Property ID values are unique throughout the entire configuration, but are automatically generated at startup based on the current configuration. The metadata packets should be used to assign Property ID values to named Properties, since most packets will only refer to a property by its ID value.

Each system has a unique system number. Each hardware device in a system has a unique control number. Each property for a control has a unique property number. They are all combined to make the 32-bit property ID value.

Offset	Length	Value	Range
0	1	System Number	0-255
1	2	Control Number	0-65535
3	1	Property Number	0-255

There may be 256 different hardware controllers, each with 65536 hardware devices, each with 256 controllable properties.

-- Matthew W Jackson - 10 Aug 2010