PosiStageNet Protocol description v2.0

© Copyright by VYV Corporation and MA Lighting Technology GmbH 2014-2016

VYV Corporation politely requests that any manufacturer who implements this protocol send details to psn@vyv.ca so that protocol updates may be communicated. Furthermore, if changes to the protocol are needed, VYV Corporation politely requests that such changes be communicated so that they may be included in the official protocol, rather than create unnecessary redundant standards.

This document defines the PosiStageNet communication specifications. This protocol was developed by VYV Corporation with input from MA Lighting International GmbH, as a means for VYV's Photon server to communicate the position of uniquely identified points in space to MA Lighting systems. Its intent is to provide a means for tracking systems to share tracking data with other show systems through an open, common and royalty-free protocol.

Communication is achieved using **UDP** for efficient and simultaneous access from multiple systems.

The PosiStageNet sends tracking information as calculated by a positioning system; it defines 2 types of packets to transmit tracking information, the **PSN_DATA** packet and the **PSN_INFO** packet. **PSN_DATA** packets are transmitted when the tracking server is active. The default transmission rate is **60Hz**, however this parameter can be modified on the tracking server and transmission could go as fast as the tracking hardware supports it (**250Hz** at the time of this writing). The **PSN_INFO** packet is transmitted at a slower rate as it is not supposed to change as often as positioning information. The default rate of **1Hz** is used for **PSN_INFO** packets.

The PosiStageNet protocol is transmitted as **UDP multicast** at address **236.10.10.10** over the port **56565**. These are the defaults parameters and can be modified on the tracking server. The protocol defines a maximum size of 1500 bytes for a single UDP packet. This limitation can cause a single frame of data to be split into multiple packets. A client must take care of reconstructing the data using the information provided in the packet header.

The PosiStageNet protocol uses a concept of CHUNKS to organize the data in a packet.

– Chunks recursively define all data sections within one data packet.

– A chunk can either contain other chunks, or binary data that needs interpretation.

– An application that is decoding a data packet and comes across unknown chunks must simply ignore them.

A PSN_CHUNK_HEADER is defined as a 32-bit unsigned integer and is structured like this:

bits 1-16	bits 17- 31	bit 32
id	data_len	has_subchunks
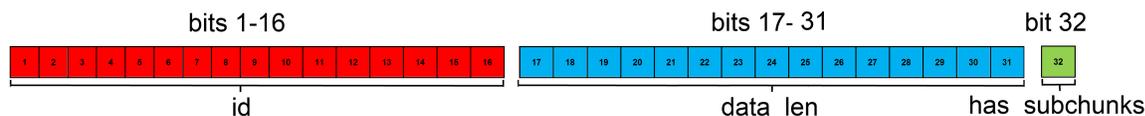
id an enum, defining the meaning of the chunk. This meaning is only valid within the current decoding context which is defined by the parent chunk.

data_len the length of the data that follows the chunk header.

has_subchunks the bit that defines, if this chunk contains child chunks.

To obtain maximum compatibility across protocol versions, care must be taken, when existing binary chunk data structures are changed:

– If a structure is expanded, new members shall be added in the end, and default values for these members must be 0. In this way, a new client can receive data from old servers by only partially reading in the data. Old clients can also receive data from newer servers by skipping the part that is too big for them.

– If such a trivial conversion is not possible or not wanted, a new id must be introduced, so old clients can safely skip the new data.

Care must be taken when introducing new ids, as different manufacturers might collide when expanding the protocol. Please write at psn@vyv.ca for any protocol change request.

DATE : Sept 15, 2016			**VYV Corporation**
			5550 Fullum suite 342
			Montréal, Québec, Canada
		®	H2G 2H4
Doc : **PosiStageNet**			514.373.2364
Protocol description	vyv.ca	**MA**	**MA Lighting Technology GmbH**
v2.02		**L I G H T I N G**	Dachdeckerstr. 16
			97297 Waldbüttelbrunn
			Germany
			Phone: +49 (0) 931 49794 -0

Data Type Definition

Type	Symbol
8-bit unsigned integer	uint8
16-bit unsigned integer	uint16
32-bit unsigned integer	uint32
64-bit unsigned integer *	uint64
32-bit float	float
32-bit float array	float[]
8-bit character	char
8-bit character array	char[]
PSN_CHUNK_HEADER	pch32

* The PosiStageNet protocol is encoded following the Little Endian convention. For integration on architectures that doesn't support 64-bit types, simply replace all 64-bit unsigned integers by two 32-bit unsigned integers. The first integer should hold the least significant bytes and the second integer the most significant bytes.

PSN_INFO packet scheme

Chunk Header	Chunk Data	Type
{0x6756} PSN_INFO_PACKET		pch32
{0x0000} PSN_INFO_PACKET_HEADER		pch32
	packet_timestamp	uint64
	version_high	uint8
	version_low	uint8
	frame_id	uint8
	frame_packet_count	uint8
{0x0001} PSN_INFO_SYSTEM_NAME		pch32
	system_name[chunk data len]	char[]
{0x0002} PSN_INFO_TRACKER_LIST		pch32
{tracker_id} PSN_INFO_TRACKER		pch32
{0x0000} PSN_INFO_TRACKER_NAME		pch32
	tracker_name[chunk data len]	char[]

PSN_INFO_PACKET
Chunk Id : 0x6756

PSN_INFO_PACKET_HEADER
Chunk Id : 0x000

Chunk Data :
uint64:   timestamp
uint8:   version_high
uint8:   version_low
uint8:   frame_id
uint8:   frame_packet_count

PSN_INFO_SYSTEM_NAME
Chunk Id : 0x0001

Chunk Data:
Char[] : system_name[chunk data len]

PSN_INFO_TRACKER_LIST
Chunk Id : 0x0002

PSN_INFO_TRACKER
Chunk Id : tracker_id

PSN_INFO_TRACKER
Chunk Id : tracker_id

PSN_INFO_TRACKER
Chunk Id : tracker_id

PSN_INFO_TRACKER_NAME
Chunk Id : 0x0000

Chunk Data :
char[] : tracker_name[chunk data len]

DATE : Sept 15, 2016

Doc : **PosiStageNet Protocol description v2.02**

vyv.ca

M A
L I G H T I N G ®

VYV Corporation
5550 Fullum suite 342
Montréal, Québec, Canada
H2G 2H4
514.373.2364

MA Lighting Technology GmbH

Dachdeckerstr. 16
97297 Waldbüttelbrunn
Germany
Phone: +49 (0) 931 49794 -0

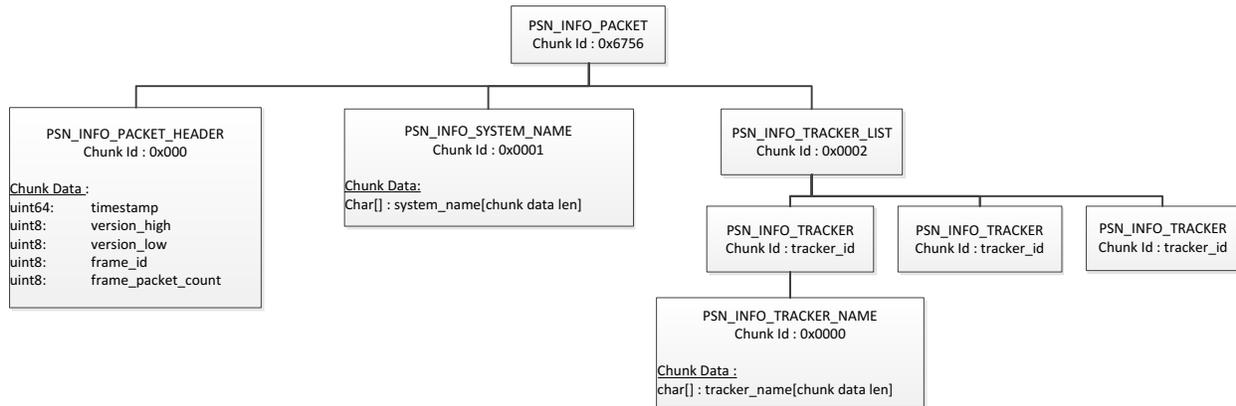## PSN_INFO_PACKET

This is the root chunk of a PSN_INFO packet.

PSN_INFO_PACKET_HEADER

timestamp	This is the number of microseconds elapsed since the PSN server was started to the moment the packet was sent by the server.
version_high	An 8-bit unsigned integer defining the high version number of the Protocol. This parameter insures that all systems using the same high version number are compatible with the sent binary packet.
version_low	An 8-bit unsigned integer defining the low version number of the Protocol.
frame_id	An 8-bit unsigned integer defining the current frame ID. A frame is a short period of time in which we analyze active trackers and give them a position and a velocity. As the UDP packet size is limited, it may happen that we need to split the information of one frame into multiple packets. The frame ID is then used to merge back the information on the client side.
frame_packet_count	An 8-bit unsigned integer defining how many packets are to be expected for the current frame ID.

PSN_INFO_SYSTEM_NAME

name	An array of characters containing the name of the PSN server.

PSN_INFO_TRACKER_LIST

This chunk is the root for all the PSN_INFO_TRACKER chunks.

PSN_INFO_TRACKER

This contains all the information of a specific tracker. The tracker Id is also the chunk Id.

PSN_INFO_TRACKER_NAME

name	An array of characters containing the name of the tracker.

PSN_DATA packet scheme

Chunk Header	Chunk Data	Type
{0x6755} PSN_DATA_PACKET		**pch32**
{0x0000} PSN_DATA_PACKET_HEADER		**pch32**
	packet_timestamp	**uint64**
	version_high	**uint8**
	version_low	**uint8**
	frame_id	**uint8**
	frame_packet_count	**uint8**
{0x0001} PSN_DATA_TRACKER_LIST		**pch32**
{tracker_id} PSN_DATA_TRACKER		**pch32**
{0x0000} PSN_ DATA_TRACKER_POS		**pch32**
	pos_x	**float**
	pos_y	**float**
	pos_z	**float**
{0x0001} PSN_ DATA_TRACKER_SPEED		**pch32**
	speed_x	**float**
	speed_y	**float**
	speed_z	**float**
{0x0002} PSN_ DATA_TRACKER_ORI		**pch_32**
	ori_x	**float**
	ori_y	**float**
	ori_z	**float**
{0x0003} PSN_ DATA_TRACKER_STATUS		**pch_32**
	validity	**float**
{0x0004} PSN_ DATA_TRACKER_ACCEL		**pch32**
	accel_x	**float**
	accel_y	**float**
	accel_z	**float**
{0x0005} PSN_ DATA_TRACKER_TRGTPOS		**pch32**
	trgtpos_x	**float**
	trgtpos_y	**float**
	trgtpos_z	**float**
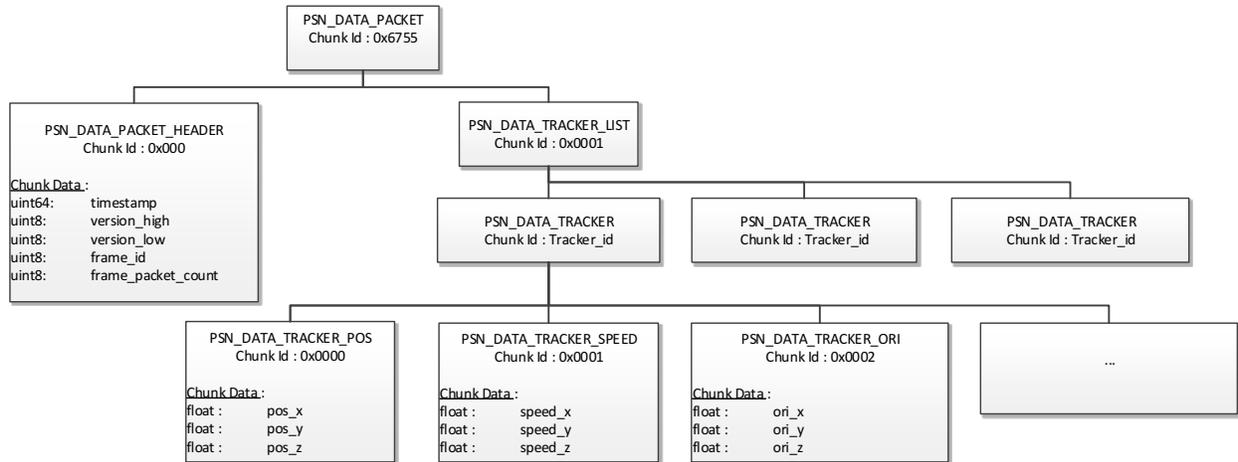
PSN_DATA_PACKET

This is the root chunk of a PSN_DATA packet.

PSN_DATA_PACKET_HEADER

timestamp	This is the number of microseconds elapsed since the PSN server was started to the moment the packet was sent by the server.
version_high	An 8-bit unsigned integer defining the high version number of the Protocol. This parameter insures that all systems using the same high version number are compatible with the sent binary packet.
version_low	An 8-bit unsigned integer defining the low version number of the Protocol.
frame_id	An 8-bit unsigned integer defining the current frame ID. A frame is a short period of time in which we analyze active trackers and give them a position and a velocity. As the UDP packet size is limited, it may happen that we need to split the information of one frame into multiple packets. The frame ID is then used to merge back the information on the client side.
frame_packet_count	An 8-bit unsigned integer defining how many packets are to be expected for the current frame ID.

PSN_DATA_TRACKER_LIST

This chunk is the root for all the PSN_DATA_TRACKER chunks.

PSN_DATA_TRACKER

This contains all the data for a specific tracker. The tracker Id is also the chunk Id.

PSN_DATA_TRACKER_POS

pos_x   A 32-bit float representing the X Euclidean Coordinates of the tracker current position. Positive x is right, positive y is up and Positive z is depth. Position is expressed in meters (m).

pos_y   A 32-bit float representing the Y Euclidean Coordinates of the tracker current position.

pos_z   A 32-bit float representing the Z Euclidean Coordinates of the tracker current position.

PSN_DATA_TRACKER_SPEED

speed_x   A 32-bit float representing the tracker current X velocity. Velocity is expressed in meters per second (m/s).

speed_y   A 32-bit float representing the tracker current Y velocity. Velocity is expressed in meters per second (m/s).

speed_z   A 32-bit float representing the tracker current Z velocity. Velocity is expressed in meters per second (m/s).

PSN_DATA_TRACKER_ORI

ori_x   A 32-bit float representing the X component of a vector indicating an axis around which the tracker is rotated. The vector's length is the amount of rotation in radians. The orientation is absolute and not cumulated from packet to packet.

ori_y   A 32-bit float representing the Y component of a vector indicating an axis around which the tracker is rotated. The vector's length is the amount of rotation in radians. The orientation is absolute and not cumulated from packet to packet.

ori_z   A 32-bit float representing the Z component of a vector indicating an axis around which the tracker is rotated. The vector's length is the amount of rotation in radians. The orientation is absolute and not cumulated from packet to packet.

PSN_DATA_TRACKER_STATUS

validity   A 32-bit float representing the tracker's validity.

PSN_DATA_TRACKER_ACCEL

accel_x   A 32-bit float representing the tracker current X acceleration. Acceleration is expressed in meters per second squared.

accel_y   A 32-bit float representing the tracker current Y acceleration. Acceleration is expressed in meters per second squared.

accel_z   A 32-bit float representing the tracker current Z acceleration. Acceleration is expressed in meters per second squared.

PSN_DATA_TRACKER_TRGTPOS

trgtpos_x   A 32-bit float representing the X position of the target that the tracker is trying to reach. Position is expressed in meters.

trgtpos _y   A 32-bit float representing the Y position of the target that the tracker is trying to reach. Position is expressed in meters.

trgtpos _z   A 32-bit float representing the Z position of the target that the tracker is trying to reach. Position is expressed in meters.