# Abreviated SeaTrac SP-48 Interface

|  |  |  |
| --- | --- | --- |
| **Date** | **Version** | **Notes** |
| 2-15-2023 | 24 | Changed definition of **Send Stream** parameter  Changed camera Start Stream command: added **Period**  Changed camera Stop Stream command: added **Send Stream**  Changed the PTZ command: added Zoom relative |

# Conventions

This document shows numbers in decimal format. So 44 shown with no other leading or trailing characters This document shows numbers in decimal format. So 44 shown with no other leading or trailing characters means decimal forty-four. In some cases it is convenient to show numbers in hexadecimal format: in those cases, the number will be shown with the C standard “0x” text beforehand, as in 0x44, which is the same as decimal 68.

# Messages

Message data to and from the SP-48 falls into several categories (Message Types), listed below.

The boat will periodically transmit the “status” messages by itself. These are periodic messages that provide status updates to the various boat systems, such as the battery system (for power updates), the GPS (for position updates), etc. The boat operator (pilot) can change the period of these updates at any time.

In the tables that follow throughout this document, the field Message Type refers to the following:

|  |  |  |
| --- | --- | --- |
| **Message Type** | **Value** | **Notes** |
| Status request | 7 | Message sent to boat requesting a specific status update |
| Status reply | 8 | Periodic status updates sent from boat |
| Request | 9 | Request to boat for a specific item (such as the current mission) |
| Reply | 10 | Reply to a specific request |
| Command | 11 | Command sent to boat |

# Connection

The standard connection to the SP-48 is from the payload computer, over the wired network, using UDP.

There are two separate connections available over UDP: the NMEA stream (by default port 62,000), and the proprietary SeaTrac protocol (by default port 62,001). This document describes the latter but be aware that there is also a very simple text-based NMEA stream that is sent from boat, and includes the data from the basic weather, GPS and AIS sensors.

To use the proprietary protocol (default port 62,001), simply open a UDP port: you should immediately see unsolicited messages coming from the SP-48.

# Data Format

Messages to the SP-48 use the following basic message structure:

|  |  |  |
| --- | --- | --- |
| **Byte(s)** | **Type** | **Notes** |
| 1 | BYTE | Sync byte 1: 0x00 |
| 2 | BYTE | Sync byte 2: 0xFF |
| 3-4 | WORD | Data Length, 0 or extra data specific to this message |
| 5 | BYTE | Message Relay – always present |
| 6 | BYTE | Message Type – always present |
| 7, 8, … | (Data) | Optional message specific data: “Data Length” bytes |
| 7 + **Data Length** | BYTE | Check1 |
| 8 + **Data Length** | BYTE | Check2 |

Some messages have no additional data: the message is a total of 8 bytes long. Other messages contain extra data: the message length is then just 8 + **Data Length**.

**Check1** and **Check2** are two checksum bytes used to check the integrity of the message. Over TCP/IP or UDP they normally wouldn’t be needed. However, on the SP-48 end, all messages are checked, and so any message sent to the SP-48 must have correct checksum bytes. Sample code to check and set the checksum is included in the section **Checksum Code** later.

## Standard Types

The following standard types are used. They are stored in memory the same way these types are stored on standard Intel x86 CPUs in C or C++ (little-endian).

|  |  |  |
| --- | --- | --- |
| **Type** | **Bytes** | **Notes** |
| BYTE | 1 | Unsigned (unsigned char) |
| WORD | 2 | Unsigned (unsigned short) |
| short | 2 | Signed (short) |
| float | 4 | Standard float |
| double | 8 | Standard double |

In some cases below, the message contains a text string. Text strings are sent as follows: the length first, then the bytes in the string. There is no null character included in the data stream.

|  |  |  |
| --- | --- | --- |
| **Byte** | **Type** | **Notes** |
| 1-2 | WORD | String length in characters, does not include null |
| 3-(string length + 2) | bytes | String (without null terminator) |

## DATETIME

This structure is used to store timestamps:

|  |  |  |
| --- | --- | --- |
| **Byte** | **Type** | **Notes** |
| 1-2 | WORD | Year: year is 1900 + WORD |
| 3 | BYTE | Month: Jan = 1, Dec = 12 |
| 4 | BYTE | Day: 1 to 31 |
| 5 | BYTE | Hour: 0 to 23 |
| 6 | BYTE | Minute: 0 to 59 |
| 7 | BYTE | Second: 0 to 59 |
| 8 | BYTE | Hundredths: 0 to 99 |

## Other Data Conversions

Some data is converted for sending to reduce traffic. In the tables that follow, you may see the following conversions:

WORDFromKts:

WORDFromDeg360:

# Standard SP-48 Status Reply Messages

The following messages are mostly status reply messages, which are updates that are sent periodically from the SP-48. The update rate can be changed by the pilot.

## Power level:

The follow message includes battery status as well as load current and pack current. Pack current is defined as (load current – charger current): negative pack current means the chargers are charging more than the load is drawing, so the pack is being charged.

|  |  |  |
| --- | --- | --- |
| **Byte(s)** | **Type** | **Value / Conversion** |
| 1 | BYTE | Sync byte 1: 0x00 |
| 2 | BYTE | Sync byte 2: 0xFF |
| 3-4 | WORD | (length) |
| 5 | BYTE | (relay) |
| 6 | BYTE | Message Type 8 (Status Reply) |
| 7 | BYTE | SinkID: 64 |
| 8-15 | DATETIME | Timestamp |
| 16-33 |  | (various) |
| 34-35 | short | Pack Current = short \* .002 |
| 36-37 | short | Load Current = short \* .002 |
| 38-39 | WORD | Pack Voltage = WORD \* .001 |
| 40-41 | WORD | SOC% = WORD \* .002 |
| 42-45 |  | (various) |
| 46 | BYTE | Check1 |
| 47 | BYTE | Check2 |

## AIS and AIS Target:

These messages contain updates to the boat’s current list of known AIS targets. The pilot can set the update rate.

This first table below describes the status reply message for the AIS system in general: it is broadcast at a regular periodic interval like the other status replies.

|  |  |  |
| --- | --- | --- |
| **Byte(s)** | **Type** | **Value / Conversion** |
| 1 | BYTE | Sync byte 1: 0x00 |
| 2 | BYTE | Sync byte 2: 0xFF |
| 3-4 | WORD | (length) |
| 5 | BYTE | (relay) |
| 6 | BYTE | Message Type 8 (Status Reply) |
| 7 | BYTE | SinkID: 26 |
| 8-15 | DATETIME | Timestamp (status update) |
| 16 | BYTE | State (0 = off, 1 = on) |
| 17 | BYTE | Number of targets |
| 18-21 | Float | Closest target (nm) |
| 22 | BYTE | Check1 |
| 23 | BYTE | Check2 |

The table below describes the reply message for an AIS target position update. **NOTE: This is not a status reply, but a reply message**. This message is not periodic – instead it is sent when the AIS target itself broadcasts a new position report (depending on operator settings, messages can be filtered to lower the update rate when the link is expensive):

|  |  |  |
| --- | --- | --- |
| **Byte(s)** | **Type** | **Value / Conversion** |
| 1 | BYTE | Sync byte 1: 0x00 |
| 2 | BYTE | Sync byte 2: 0xFF |
| 3-4 | WORD | (length) |
| 5 | BYTE | (relay) |
| 6 | BYTE | Message Type 10 (Reply) |
| 7 | BYTE | BoardID: 1 |
| 8 | BYTE | SinkID: 26 |
| 9 | BYTE | Function: 2 |
| 10-17 | DATETIME | Timestamp (position report) |
| 18 | double | Latitude (radians) |
| 19 | double | Longitude (radians) |
| 20-23 | DWORD | MMSI |
| 24-25 | WORD | Speed over ground = .002 \* WORD kts |
| 26-27 | WORD | Course over ground = .01 \* WORD |
| 28-29 | WORD | Heading |
| 30 | BYTE | Navigation Status |
| 31 | BYTE | Check1 |
| 32 | BYTE | Check2 |

The table below is similar to the position update, but it describes the reply message for AIS target details update. **NOTE: As above, this is not a status reply, but a reply**. These are sent much less frequently.

|  |  |  |
| --- | --- | --- |
| **Byte(s)** | **Type** | **Value / Conversion** |
| 1 | BYTE | Sync byte 1: 0x00 |
| 2 | BYTE | Sync byte 2: 0xFF |
| 3-4 | WORD | (length) |
| 5 | BYTE | (relay) |
| 6 | BYTE | Message Type 10 (Reply) |
| 7 | BYTE | BoardID: 1 |
| 8 | BYTE | SinkID: 26 |
| 9 | BYTE | Function: 4 |
| 10-13 | DWORD | MMSI |
| 14 | BYTE | Ship type |
| 15-36 | string | Ship name, up to 20 chars long |
| 37-45 | string | Call sign, up to 7 chars long |
| 46-47 | WORD | M to bow |
| 48-49 | WORD | M to stern |
| 50 | BYTE | M to port |
| 51 | BYTE | M to starboard |
| 52 | BYTE | Check1 |
| 53 | BYTE | Check2 |

## Propeller Motor, Steering Motor:

These messages provide more detail about the low-level motor state: target and actual RPM, temperature of the amplifier, power usage, etc. We can provide more detail if these are useful in the future.

## Attitude and GPS (and Backup):

The attitude of the boat (heading, pitch, roll) and the position of the boat (from the GPS) is tracked by two independent sensors. The table below describes the messages that come from these two sensors: the messages are the same, except for the SinkID. For the attitude, the main sensor is 76 and the backup is 79. For the GPS, the main sensor is 77 and the backup is 80.

The table below describes the message for **attitude**:

|  |  |  |
| --- | --- | --- |
| **Byte(s)** | **Type** | **Value / Conversion** |
| 1 | BYTE | Sync byte 1: 0x00 |
| 2 | BYTE | Sync byte 2: 0xFF |
| 3-4 | WORD | (length) |
| 5 | BYTE | (relay) |
| 6 | BYTE | Message Type 8 (Status reply) |
| 7 | BYTE | SinkID: 76, backup 79 |
| 8-15 | DATETIME | Timestamp (attitude report) |
| 16-17 | WORD | Heading = .01 \* WORD |
| 18-19 | Short | Pitch = .01 \* short |
| 20-21 | Short | Min pitch = .01 \* short |
| 22-23 | Short | Max pitch = .01 \* short |
| 24-25 | Short | Roll = .01 \* short |
| 26-27 | Short | Min roll = .01 \* short |
| 28-29 | Short | Max roll = .01 \* short |
| 30-31 | Short | Min heading = .01 \* WORD |
| 32-33 | Short | Max heading = .01 \* WORD |
| 34 | BYTE | Check1 |
| 35 | BYTE | Check2 |

The table below describes the message for **GPS** position:

|  |  |  |
| --- | --- | --- |
| **Byte(s)** | **Type** | **Value / Conversion** |
| 1 | BYTE | Sync byte 1: 0x00 |
| 2 | BYTE | Sync byte 2: 0xFF |
| 3-4 | WORD | (length) |
| 5 | BYTE | (relay) |
| 6 | BYTE | Message Type 8 (Status reply) |
| 7 | BYTE | SinkID: 77, backup 80 |
| 8-15 | DATETIME | Timestamp (position report) |
| 16-23 | double | Latitude (radians) |
| 24-31 | double | Longitude (radians) |
| 32-33 | WORD | Kts = .002 \* WORD kts |
| 34-35 | WORD | Heading = .01 \* WORD |
| 36-37 | WORD | Current kts = .002 \* WORD kts |
| 38-39 | WORD | Current hdg = .01 \* WORD |
| 40-41 | WORD | Wind kts = .002 \* WORD kts |
| 42-43 | WORD | Wind hdg = .01 \* WORD |
| 44 | BYTE | Check1 |
| 45 | BYTE | Check2 |

## IMU:

Separate from the attitude and GPS sensors described above, the SP-48 has an internal IMU that is able to get higher frequency. It can also estimate heave, and thus provides an estimate of wave height. The table below describes the data from the IMU:

|  |  |  |
| --- | --- | --- |
| **Byte(s)** | **Type** | **Value / Conversion** |
| 1 | BYTE | Sync byte 1: 0x00 |
| 2 | BYTE | Sync byte 2: 0xFF |
| 3-4 | WORD | (length) |
| 5 | BYTE | (relay) |
| 6 | BYTE | Message Type 8 (Status reply) |
| 7 | BYTE | SinkID: 86 |
| 8-15 | DATETIME | Timestamp (attitude) |
| 16-17 | short | Roll = .01 \* short |
| 18-19 | short | Min roll = .01 \* short |
| 20-21 | short | Max roll = .01 \* short |
| 22-23 | short | Pitch = .01 \* short |
| 24-25 | short | Min pitch = .01 \* short |
| 26-27 | short | Max pitch = .01 \* short |
| 28-29 | WORD | Heading = .01 \* WORD |
| 30-31 | short | Roll gyro rate = .02 \* short deg / sec |
| 32-33 | short | Pitch gyro rate = .02 \* short deg / sec |
| 34-35 | short | Heading gyro rate = .02 \* short deg / sec |
| 36-37 | short | Acceleration in X (forward) = .01 \* short m/s^2 |
| 38-39 | short | Acceleration in Y (starboard) = .01 \* short m/s^2 |
| 40-41 | short | Acceleration in Z (down) = .01 \* short m/s^2 |
| 42-43 | short | Max acceleration in X = .01 \* short m/s^2 |
| 44-45 | short | Max acceleration in Y = .01 \* short m/s^2 |
| 46-47 | short | Max acceleration in Z = .01 \* short m/s^2 |
| 48-49 | short | Z (heave) = .001 \* short m |
| 50-51 | short | Minimum Z = .001 short m |
| 52-53 | short | Maximum Z = .001 short m |
| 54-55 | WORD | Minimum heading = .01 \* WORD |
| 56-57 | WORD | Maximum heading = .01 \* WORD |
| 58 | BYTE | Check1 |
| 59 | BYTE | Check2 |

## Wind:

This data includes the apparent wind speed and direction, plus air temperature and air pressure. The table below describes the data from the wind sensor:

|  |  |  |
| --- | --- | --- |
| **Byte(s)** | **Type** | **Value / Conversion** |
| 1 | BYTE | Sync byte 1: 0x00 |
| 2 | BYTE | Sync byte 2: 0xFF |
| 3-4 | WORD | (length) |
| 5 | BYTE | (relay) |
| 6 | BYTE | Message Type 8 (Status reply) |
| 7 | BYTE | SinkID: 78 |
| 8-15 | DATETIME | Timestamp |
| 16-17 | WORD | Apparent speed = .002 \* WORD kts |
| 18-19 | short | Apparent angle = .01 \* short |
| 20-21 | short | Temperature = .01 \* short degC |
| 22-23 | short | Pressure = .01 \* short Bar |
| 24 | BYTE | Check1 |
| 25 | BYTE | Check2 |

## Propulsion and Navigator:

These messages provide data about the boat’s propulsion and navigation. This includes the boat’s “state” – such as “drifting”, “holding position”, and “going to a waypoint”. It also includes the target RPM and heading as well as the two active waypoints on the target track line (when the boat is doing a mission).

The table below describes the message from the propulsion system:

|  |  |  |
| --- | --- | --- |
| **Byte(s)** | **Type** | **Value / Conversion** |
| 1 | BYTE | Sync byte 1: 0x00 |
| 2 | BYTE | Sync byte 2: 0xFF |
| 3-4 | WORD | (length) |
| 5 | BYTE | (relay) |
| 6 | BYTE | Message Type 8 (Status reply) |
| 7 | BYTE | SinkID: 82 |
| 8-15 | DATETIME | Timestamp |
| 16-17 | short | Propulsion state (see below) |
| 18-19 | short | Target RPM |
| 20-21 | short | Target rudder angle = .01 \* short deg |
| 22-23 | WORD | Target heading = WORD \* .01 |
| 24-25 | short | (reserved) |
| 26-27 | short | (reserved) |
| 28-29 | short | (reserved) |
| 30-31 | short | Actual rudder angle = .01 \* short deg |
| 32-33 | short | Actual rudder speed = .01 \* short deg / sec |
| 34 | BYTE | Check1 |
| 35 | BYTE | Check2 |

**State** identifies the propulsion state:

|  |  |
| --- | --- |
| **State** | **State** |
| 0 | Unknown |
| 1 | Off |
| 2 | Power on |
| 3 | Enabling motors |
| 4 | Idle |
| 5 | Homing steering motor |
| 6 | Ready (Drift in user interface) |
| 7 | Rudder angle and RPM control |
| 8 | Heading and RPM control |
| 9 | Track and RPM control |

The table below describes the message from the navigation system. Note that this message changes depending on what the boat is doing: for example, when the boat is executing a mission, additional items are added to the message with details about the current track and alternate waypoint or heading, if active.

|  |  |  |
| --- | --- | --- |
| **Byte(s)** | **Type** | **Value / Conversion** |
| 1 | BYTE | Sync byte 1: 0x00 |
| 2 | BYTE | Sync byte 2: 0xFF |
| 3-4 | WORD | (length) |
| 5 | BYTE | (relay) |
| 6 | BYTE | Message Type 8 (Status reply) |
| 7 | BYTE | SinkID: 83 |
| 8-15 | DATETIME | Timestamp |
| 16-17 | short | Mission state (see below) |
| 18-19 | WORD | Requested speed = .002 \* WORD kts |
| 20-21 | short | Target RPM |
| 22-23 | WORD | N Waypoints |
| 24-25 | WORD | Next waypoint index (0xFFFF if none) |
| If next wp valid | double | Leg start latitude (radians) |
| If next wp valid | double | Leg start longitude (radians) |
| If next wp valid | double | Leg end latitude (radians) |
| If next wp valid | double | Leg end longitude (radians) |
|  | WORD | Laps to do (0 for single shot mission) |
|  | WORD | Laps done |

If the boat is holding (mission state is 6, see below), then the next bytes are:

|  |  |  |
| --- | --- | --- |
|  | double | Latitude (radians) |
|  | double | Longitude (radians) |
|  | float | Hold Rin ft |
|  | float | Hold Rout ft |
|  | float | Hold kts |

If the boat is running a mission (mission state is 7, see below), then:

|  |  |  |
| --- | --- | --- |
|  | DWORD | Holding at waypoint: time remaining secs |
| if hold seconds != 0 | float | Holding at waypoing: Rout ft |
|  | bool | Alternate waypoint active |
| if alternate waypoint | double | Alternate waypoint latitude (radians) |
| if alternate waypoint | double | Alternate waypoint longitude (radians) |
|  | bool | Alternate heading active |
| if alternate heading | float | Alternate heading |
|  | bool | Alternate speed active |
| if alternate kts | float | Alternate speed kts |

Then the message ends with standard checksum:

|  |  |  |
| --- | --- | --- |
|  | BYTE | Check1 |
|  | BYTE | Check2 |

**State** identifies the Navigator state:

|  |  |
| --- | --- |
| **State** | **State** |
| 0 | Unknown |
| 1 | Off |
| 2 | Starting |
| 3 | Drift |
| 4 | RC manual mode |
| 5 | RC auto mode |
| 6 | Hold position |
| 7 | Mission |
| 8 | All stop (emergency stop) |

# Messages for Video Data

The following sections describe the messages and commands that relate to the video cameras.

The camera streams can be started and stopped. When a stream is started, the command tells the boat what resolution to use: low, medium or high. The command also tells the boat to send images: 1) continuously at the standard frame rate (10 fps), 2) at a slower periodic rate (such as one frame every 30 seconds), or 3) just one frame. The lower resolution and lower frame rates are used when the data link is expensive, or the link has too much traffic and there is a desire to limit the upload bandwidth.

The boat uploads image data in reply messages using the regular message format. The image data is in the H.264, and includes messages for: 1) setup frames to tell the decoder about the frame size etc, 2) key frames that contain all the data for an image, and 3) intermediate frames which have data to update a previous image. Intermediate frames are only used when streaming continuously: they are not used for single or periodic image capture.

Note that H.264 image packets typically don’t fit into a single SP-48 message. Therefore, H.264 data packets are broken into “chunks” and then reassembled on receipt.

H.264 setup frames are always sent in advance of key frames, since the data link can go down, and resolution can change between frames. Setup frames are small and will always be contained by themselves in one SP-48 message.

The attitude of the boat (heading, pitch and roll) is also sent along with the video data. This enables the receiver to know where the camera was aimed when the image was captured, and for PTZ cameras, the azimuth angle, the tilt angle and the zoom level. As described below, the attitude for each frame is sent as part of the first image chunk.

The next sections describe the image data sent by the SP-48. Later, the commands to control the camera are covered.

## Image Data Start:

All image data sent from the SP-48 has the following starting sequence:

|  |  |  |
| --- | --- | --- |
| **Byte(s)** | **Type** | **Value / Conversion** |
| 1 | BYTE | Sync byte 1: 0x00 |
| 2 | BYTE | Sync byte 2: 0xFF |
| 3-4 | WORD | (length) |
| 5 | BYTE | (relay) |
| 6 | BYTE | Message Type: 10 (Reply) |
| 7 | BYTE | BoardID: 10 |
| 8 | BYTE | SinkID: 95 |
| 9 | BYTE | Function: 5 |
| 10 | BYTE | Location (see below) |
| 11 | BYTE | Send Stream (see below) |
| 12-13 | WORD | NChunks |
| 14-15 | WORD | Chunk (this chunk) |

**Location** identifies the camera stream:

|  |  |
| --- | --- |
| **Location** | **Camera** |
| 0 | Bow |
| 1 | Port |
| 2 | Starboard |
| 3 | Stern |
| 4 | PTZ – IR |
| 5 | PTZ – EO |

**Send Stream** is a bitfield to identify the link(s) used to upload the data:

|  |  |
| --- | --- |
| **Send Stream** | **Link** |
| 0 | None |
| 1 | Cell Link |
| 2 | Certus Link |
| 4 | Silvus Link |

## H.264 Setup Frame Messages:

As mentioned above, the boat will send H.264 setup frames before each key frame: there are typically two setup frames (so two messages). All setup frames should be sent to the decoder. Setup frames will always have **NChunks** = 1 and **This chunk** = 0.

The table below lists the data that follows the Image Data Start for setup frames:

|  |  |  |
| --- | --- | --- |
| **Byte(s)** | **Type** | **Value / Conversion** |
| 16 – 17 | WORD | Setup frame size (bytes) |
| 18 – (Data Length + 6) | (data) | Setup frame data |
| (Data Length + 7) | BYTE | Check1 |
| (Data Length + 8) | BYTE | Check2 |

Be sure to send all setup frames to the decoder before sending real image data to the decoder.

## H.264 Image Frame Messages: Chunk = 1:

When the boat sends image data (key frames or intermediate frames), it will typically break those messages into chunks because the data won’t all fit into one message. The first **Chunk** is 1 (not 0), and the last **Chunk** is **NChunks**. Both **Chunk** and **NChunks** are set in each message as described above.

The first **Chunk** (**Chunk** = 1) contains the boat attitude and must be parsed differently than the remaining chunks. The table below picks up at byte 16 and lists the remaining data in the message for chunk 1:

|  |  |  |
| --- | --- | --- |
| **Byte(s)** | **Type** | **Value / Conversion** |
| 16 – 19 | float | Strength (for distortion, not used on PTZ) |
| 20 – 23 | float | Zoom (for distortion, not used on PTZ) |
| 24 – 27 | float | Height of camera above water ft |
| 28 – 31 | float | Heading (the boat’s heading) |
| 32 – 35 | float | PTZ azimuth angle |
| 36 – 39 | float | PTZ elevation angle |
| 40 – 43 | float | PTZ view angle (total horizontal field of view) |
| 44 – 47 | float | Image rotation (for correction, not used on PTZ) |
| 48 – 51 | float | Image height (for correction, not used on PTZ) |
| 52 – 59 | DATETIME | Image timestamp |
| 60 - 61 | WORD | Image data size (bytes) |
| 62 - (Data Length + 6) | (data) | Image data |
| (Data Length + 7) | BYTE | Check1 |
| (Data Length + 8) | BYTE | Check2 |

## H.264 Image Frame Messages: Chunk > 1:

The chunks that follow (**Chunk** = 2, 3, … **NChunks)** have the following data:

|  |  |  |
| --- | --- | --- |
| **Byte(s)** | **Type** | **Value / Conversion** |
| 16 – 17 | WORD | Image data size (bytes) |
| 18 – (Data Length + 6) | (data) | Image data |
| (Data Length + 7) | BYTE | Check1 |
| (Data Length + 8) | BYTE | Check2 |

Reassemble the chunks by appending the data in chunk 2 to the data in chunk 1, and so on until all chunks have been received. Then send the reassembled image data to the decoder.

# Camera Commands

## Start Stream:

This message commands the SP-48 to start a camera stream. It can also be used to restart a stream that is already running, but with new settings.

|  |  |  |
| --- | --- | --- |
| **Byte(s)** | **Type** | **Value / Conversion** |
| 1 | BYTE | Sync byte 1: 0x00 |
| 2 | BYTE | Sync byte 2: 0xFF |
| 3-4 | WORD | (length) |
| 5 | BYTE | Relay: 0 |
| 6 | BYTE | Message Type: 11 (Command) |
| 7 | BYTE | BoardID: 10 |
| 8 | BYTE | SinkID: 95 |
| 9 | BYTE | Function: 2 |
| 10 | BYTE | Location |
| 11 | BYTE | Resolution (see below) |
| 12 | WORD | Period (see below) |
| 13 | BYTE | Send Stream |
| 14 | BYTE | Check1 |
| 15 | BYTE | Check2 |

**Location** and **Send Stream** are defined above. **Resolution** is listed below:

|  |  |
| --- | --- |
| **Resolution** | **Resolution** |
| 0 | Low (360p) |
| 1 | Medium (720p) |
| 2 | High (1080p) |

The **Period** is used to specify a continuous stream, or a period frame update, or a single still image.

|  |  |
| --- | --- |
| **Period** | **Frame Timing** |
| 0 | Stream (continuous) |
| 1 - 998 | Seconds between frames |
| 999 | Single still image |

## Stop Stream:

This message commands the SP-48 to stop a stream, by specifying the camera location as well as the link you want to stop. Use **Location** and **Send Stream** as defined above.

|  |  |  |
| --- | --- | --- |
| **Byte(s)** | **Type** | **Value / Conversion** |
| 1 | BYTE | Sync byte 1: 0x00 |
| 2 | BYTE | Sync byte 2: 0xFF |
| 3-4 | WORD | (length) |
| 5 | BYTE | Relay: 0 |
| 6 | BYTE | Message Type: 11 (Command) |
| 7 | BYTE | BoardID: 10 |
| 8 | BYTE | SinkID: 95 |
| 9 | BYTE | Function: 3 |
| 10 | BYTE | Location |
| 11 | BYTE | Send Stream |
| 12 | BYTE | Check1 |
| 13 | BYTE | Check2 |

This message stops all streams.

|  |  |  |
| --- | --- | --- |
| **Byte(s)** | **Type** | **Value / Conversion** |
| 1 | BYTE | Sync byte 1: 0x00 |
| 2 | BYTE | Sync byte 2: 0xFF |
| 3-4 | WORD | (length) |
| 5 | BYTE | Relay: 0 |
| 6 | BYTE | Message Type: 11(Command) |
| 7 | BYTE | BoardID: 10 |
| 8 | BYTE | SinkID: 95 |
| 9 | BYTE | Function: 4 |
| 10 | BYTE | Check1 |
| 11 | BYTE | Check2 |

## PTZ:

The following commands are used to turn the PTZ camera. Note that the turning of the camera is independent of the video streams, so commands to start and stop streams do not impact commands to move the camera, just as commands to move the camera do not impact the streams.

Note also that the IR and EO cameras in the PTZ are physically in the same housing, so both cameras will pan and tilt to the same location.

Additionally, the IR camera does not have a lens for zooming, so zoom commands will only impact the EO stream.

|  |  |  |
| --- | --- | --- |
| **Byte(s)** | **Type** | **Value / Conversion** |
| 1 | BYTE | Sync byte 1: 0x00 |
| 2 | BYTE | Sync byte 2: 0xFF |
| 3-4 | WORD | (length) |
| 5 | BYTE | Relay: 0 |
| 6 | BYTE | Message Type: 11 (Command) |
| 7 | BYTE | BoardID: 10 |
| 8 | BYTE | SinkID: 95 |
| 9 | BYTE | Function: 5 |
| 10 | BYTE | PTZ command |
| 11 | short | PTZ parameter |
| 12 | BYTE | Check1 |
| 13 | BYTE | Check2 |

PTZ commands rely on the **PTZ command** and **PTZ parameter**, as listed below:

|  |  |  |  |
| --- | --- | --- | --- |
| **PTZ Command** | **PTZ Parameter** | **Command** | **Notes** |
| 0 | Pan absolute | Pan to absolute angle | .1 deg, 0 degrees is forward, range is +/- 1800 |
| 1 | Pan relative | Pan to relative angle | .1 deg, range is +/- 1800 |
| 2 | Pan jogging | Set pan jog speed | %, 0 is stop, negative to port, range is +/- 100 |
| 3 | Tilt absolute | Tilt to absolute angle | .1 deg, 0 degrees is level, range is +/- 900 |
| 4 | Tilt relative | Tilt to relative angle | .1 deg, range is +/- 900 |
| 5 | Tilt jogging | Start tilt jogging | %, 0 is stop, negative down, range is +/- 100 |
| 6 | Zoom | Zoom to absolute % | %, 0 is zoomed out, range is 0 to 100 |
| 7 | Zoom relative | Zoom to relative % | %, negative is out, range is +/- 100 |
| 8 | Zoom jogging | Start zoom jogging | %, 0 is stop, negative out, range is +/- 100 |
| 8 | Stabilize | Set camera stabilization | 0: none, 1: pitch only, 2: pitch and yaw |

Here are some examples the help illustrate.

Pan to absolute 0: this moves the camera so it points forward, it is very quick (a second or so). This is useful if the operator wants to quickly see what is forward. Zoom absolute 0 is also useful to quickly zoom out for a full view.

Pan jogging at 5%: this will turn the camera at a fixed 5% of max speed, so the operator sees a sweeping view of the horizon around the boat. A subsequent command of Pan jogging 0% stops the camera.

Zoom jogging at 10%: this zooms the lens in at 10% so the operator can see more detail. A subsequent command of zoom jogging 0% stops the zooming, or a zoom jogging -10% will zoom slowly back out.

Stabilize set to 1 (pitch) is very useful to minimize the impact of boat pitch and roll on the image. Note that with stabilize set to 1, the camera will follow the boat as it turns: if the camera is aimed forward, it will remain pointed forward as the boat turns.

With stabilize set to 2, the camera will correct for yaw in addition to the pitch / roll. This is useful if the operator wants to track a fixed object but the SP-48 is turning or needs to turn.

# Other SP-48 Commands

This section lists messages that command the SP-48 to do various things unrelated to the cameras.

## Change State:

The state of the boat is best checked by looking at the Navigator: use the status reply message to determine what state it’s in. Likewise, the way to change the state of the boat is also through the Navigator.

The message described below requests a state change: note that some state changes take time (powering up, homing motors, etc): monitor the state being returned in the Navigator status reply to see if the desired state has been reached.

On power up, the system must first go to Drift, before attempting any commanded moves.

The following table describes the command to request a Navigator state change:

|  |  |  |
| --- | --- | --- |
| **Byte(s)** | **Type** | **Value / Conversion** |
| 1 | BYTE | Sync byte 1: 0x00 |
| 2 | BYTE | Sync byte 2: 0xFF |
| 3-4 | WORD | (length) |
| 5 | BYTE | Relay: 8 (internal CAN bus) |
| 6 | BYTE | Message Type: 11 (Command) |
| 7 | BYTE | BoardID 8 |
| 8 | BYTE | SinkID: 83 |
| 9 | BYTE | Function: 5 |
| 10-13 | int | New state (see table from status replies – same values) |
| 14 | BYTE | Check1 |
| 15 | BYTE | Check2 |

## Rudder Angle and RPM:

To use the rudder angle and RPM commands, the Navigator must first be put into the RC Manual state. Once in that state, the table below describes the message to call for the desired rudder angle and RPM:

|  |  |  |
| --- | --- | --- |
| **Byte(s)** | **Type** | **Value / Conversion** |
| 1 | BYTE | Sync byte 1: 0x00 |
| 2 | BYTE | Sync byte 2: 0xFF |
| 3-4 | WORD | (length) |
| 5 | BYTE | Relay: 8 (internal CAN bus) |
| 6 | BYTE | Message Type: 11 |
| 7 | BYTE | BoardID: 8 |
| 8 | BYTE | SinkID: 82 |
| 9 | BYTE | Function: 1 |
| 10-13 | float | Rudder angle deg (0 is straight, positive values turn to starboard) |
| 14-17 | float | RPM (see table below) |
| 18 | BYTE | Check1 |
| 19 | BYTE | Check2 |

|  |  |
| --- | --- |
| **RPM** | **Speed (kts)** |
| 0 | 0 |
| 200 | .7 |
| 400 | 1.6 |
| 500 | 2 |
| 600 | 2.5 |
| 700 | 2.9 |
| 800 | 3.3 |
| 900 | 3.6 |
| 1000 | 4 |
| 1100 | 4.3 |
| 1200 | 4.6 |
| 1300 | 5 |

## Heading and RPM:

To use the heading and RPM commands, the Navigator must first be put into the RC Auto state. Once in that state, the table below describes the message to call for the desired heading and RPM:

|  |  |  |
| --- | --- | --- |
| **Byte(s)** | **Type** | **Value / Conversion** |
| 1 | BYTE | Sync byte 1: 0x00 |
| 2 | BYTE | Sync byte 2: 0xFF |
| 3-4 | WORD | (length) |
| 5 | BYTE | Relay: 8 (internal CAN bus) |
| 6 | BYTE | Message Type: 11 (Command) |
| 7 | BYTE | BoardID: 8 |
| 8 | BYTE | SinkID: 82 |
| 9 | BYTE | Function: 2 |
| 10-13 | float | Heading (desired true heading, not magnetic) |
| 14-17 | float | RPM |
| 18 | BYTE | Check1 |
| 19 | BYTE | Check2 |

## Move to Point:

The follow command message instructs the SP-48 to move to a new location:

|  |  |  |
| --- | --- | --- |
| **Byte(s)** | **Type** | **Value / Conversion** |
| 1 | BYTE | Sync byte 1: 0x00 |
| 2 | BYTE | Sync byte 2: 0xFF |
| 3-4 | WORD | (length) |
| 5 | BYTE | Relay: 8 (internal CAN bus) |
| 6 | BYTE | Message Type: 11 (Command) |
| 7 | BYTE | BoardID: 8 |
| 8 | BYTE | SinkID: 83 |
| 9 | BYTE | Function: 7 |
| 10-17 | double | Latitude (radians) |
| 18-25 | double | Longitude (radians) |
| 26-29 | float | Speed (kts) |
| 30 | BYTE | Check1 |
| 31 | BYTE | Check2 |

# Checksum Code

The following code shows how to check the checksum bytes:

BYTE \*pData = (Buffer with message: starting with first byte (0x00))

int iDataLength = (length of message specific data)

int iHeaderLength = 6;

int iSum1 = 0, iSum2 = 0;

int iLength = iHeaderLength + iDataLength;

while (iLength--)

{

iSum1 += \*pData++;

if (iSum1 >= 255)

iSum1 -= 255;

iSum2 += iSum1;

if (iSum2 >= 255)

iSum2 -= 255;

}

ASSERT((iSum1 <= 255) && (iSum2 <= 255));

int iCheck1 = 255 - (iSum1 + iSum2) % 255;

int iCheck2 = 255 - (iSum1 + iCheck1) % 255;

BYTE byCheck1 = \*pData++;

BYTE byCheck2 = \*pData;

if (byCheck1 != (BYTE) iCheck1)

return false;

if (byCheck2 != (BYTE) iCheck2)

return false;

return true;

The following code shows how to set the checksum bytes:

BYTE \*pData = (Buffer with message: starting with first byte (0x00))

int iDataLength = (length of message specific data)

int iHeaderLength = 6;

int iSum1 = 0, iSum2 = 0;

int iLength = iHeaderLength + iDataLength;

while (iLength--)

{

iSum1 += \*pData++;

if (iSum1 >= 255)

iSum1 -= 255;

iSum2 += iSum1;

if (iSum2 >= 255)

iSum2 -= 255;

}

ASSERT((iSum1 <= 255) && (iSum2 <= 255));

int iCheck1 = 255 - (iSum1 + iSum2) % 255;

int iCheck2 = 255 - (iSum1 + iCheck1) % 255;

\*pData++ = (BYTE) iCheck1;

\*pData = (BYTE) iCheck2;