icListen Streaming Telemetry

October 24, 2018

Ocean Sonics Ltd.
Hill House, 11 Lornevale Road,
Great Village, NS, B0M 1L0 Canada
Phone: +1 902 655 3000
www.OceanSonics.com

Table of Contents

1 Introduction

This document applies to the following Ocean Sonics products:

icListen HF v2.2 (Release 34) and below
icListen AF v2.2 (Release 34) and below
Host Controller (Release 1)

If your device is not included in the above list or is a higher firmware version than those listed above, please contact Ocean Sonics for the latest revision of this document.

This document details the telemetry format for communicating with an **icListen** unit over the streaming channels. **icListen AF**, **icListen HF** and the **Host Controller** support data streaming channels. All **icListen** models support a command & control channel.

Data may be streamed via the UDP or TCP network protocols. Each requested data type (wave data, FFT data, etc.) will be sent on a separate port. All data within streaming messages is in network order (big endian), except for the waveform data body which is the configured data format.

Data collection and processing settings cannot be configured, using the streaming channels. To accomplish this, the command and control channel should be used. For more information on the command and control channel telemetry, please see the *icListen Command & Control Telemetry* document.

2 Overview of Messages

This section contains a brief overview of the messages used by **icListen** stream channels. All message payloads must be 32bit aligned and are in network order (big endian) unless otherwise noted.

2.1 Basic Message Structure

All messages transmitted over **icListen**'s streaming channels (except for the epoch message stream, which sends ASCII data) share a common structure:

Byte Offset:	Field:	# Bytes:
0	Message Type	1
1	Sync (0x2A)	1
2	Payload Length	2
4	Payload	n

Figure 2-1: Basic Message Structure

Message Type: This byte identifies the actual type of message being transmitted.
Sync: The second character in all messages must be the sync character. This character is an ASCII '*' character (0x2A).
Payload Length: This is the number of bytes in the payload of the message. A payload length of 0 is valid.
Payload: The payload will vary in length and content, based on the type of message being transmitted.

2.2 Streaming Channel Messages

Message	***Code***	***Description***
Notify	'0' (0x30)	Sends important notifications
Data	'1' (0x31)	Data transmitted from icListen/Host Controller
Event Header	'2' (0x32)	Contains header information related to the following data messages
Start Stream	'3' (0x33)	Requests a data stream be opened
Stop Stream	'4' (0x34)	Requests a data stream be closed
Other Sensor Data	'5' (0x35)	Data from other sensors on the hydrophones (Temp, Humidity, etc)

2.3 Streamed Data Behavior

Streamed message channels all transmit *Data* messages, containing the requested data for that channel. Waveform and spectrum channels will also transmit an *Event Header* once every second, before any data corresponding to that second, which contain data handling parameters. The event header messages are sent only once per second to reduce bandwidth used when sending setup information which will not change within the second. Multiple data messages may be sent between each event header message. The message ordering is depicted below.

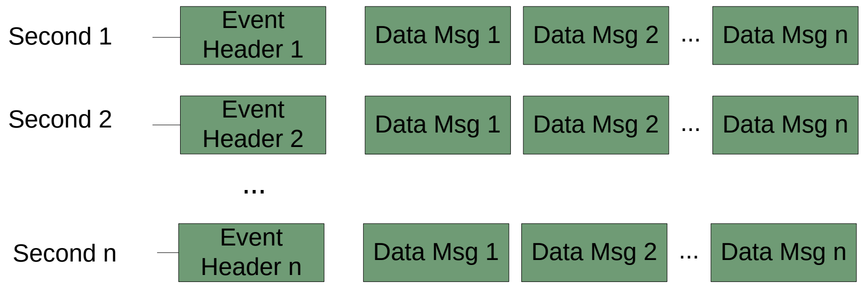

Figure 2-2: Message Ordering

The payloads of a both the event and data messages are broken into chunks. For more information on the chunks used to make up these messages, please refer to the *Message Chunk Details* section. After a stream has been activated, it will remain active until either: the time the stream was requested for expires, a *Stop Stream* message is received, a *Stop All Streams* message is received over the command and control channel, or a lost connection has been detected.

2.4 Available Streaming Channels

icListen makes use of different channels for each type of data to be streamed. Currently **icListen** supports streaming of waveform, spectrum, epoch message, and other sensor data over TCP. Additionally, waveform and spectrum data may also be streamed over UDP. The following table shows what streaming channels are available:

Streaming Channels:

Data Type	**Protocol**	**Port**
Time Series (Waveform)	UDP	51676
	TCP	51678
Spectrum (FFT)	UDP	51677
	TCP	51679
Epoch Messages	TCP	51680
Other Sensors	TCP	51681

2.5 Epoch Message Stream

As of Release 15, **icListen AF/HF** can stream epoch messages whenever an epoch event is triggered, and when the trigger condition goes away. The epoch stream is only available over TCP. For the epoch stream, connect to port 51680, and the unit will send epoch messages whenever an epoch trigger which is configured to send the epoch messages occurs and (as of Release 19) when a trigger ends.
The epoch messages are ASCII strings.

Example trigger activated string:

```
  2012-09-20 10:39:55,U 1219,Epoch 1,Seq 36430802944,Hz 10000-25000 > 100uPa,3sec
```

The format is:

```
  [Time],U [Serial #],Epoch [Epoch #],Seq [Sequence #],Hz [Freq. Range] [> or <] [Amplitude],[Duration]
```

Example trigger ended string:

```
  2012-09-20 10:39:55,U 1219,Epoch 1,Ended
```

The format is:

```
  [Time],U [Serial #],Epoch [Epoch #],Ended
```

For both strings, time is in the form: yyyy-mm-dd hh:mm:ss

2.6 Other Sensor Data Stream

As of release 28, **icListen AF/HF** can stream data from other sensors on the hydrophone. The other sensors which can send data through this stream are temperature, humidity, battery details, time sync/PPS, accelerometer and magnetometer, and trigger status. The other sensor stream is only available over TCP. For the other sensor stream, connect to port 51681, and the unit will send a message whenever sensor data is updated. Data will only be transmitted from the sensors requested in the start stream message.

3 Detailed Message Descriptions

This section gives detailed descriptions of all messages transmitted over the streaming data channels. These messages used by all channels, except for the epoch message channel, which does not accept an messages, and transmits ASCII data.

The only messages accepted into the **icListen** streaming ports are those required to start and stop streaming:

Start Stream
Stop Stream

icListen will transmit the following messages over the streaming channels:

Notify
Event Header
Data
Other Sensor Data

3.1 Start Stream

The purpose of this message is to start streaming data for the requested amount of time. It is required by all binary data streams. Sending start stream messages after a stream is active, will override the duration and requested data fields for the currently active stream, without closing off the channel and without interrupting data transmission.

Byte Offset:	Field:	# Bytes:
0	Type '3' (0x33)	1
1	Sync (0x2A)	1
2	Payload Length	2
4	Duration	2
6	Requested Data	2

Figure 3-1: Start Message

Duration: This is the length of time (in minutes) to stream data from the device. A value of 0 will cause the stream to remain active indefinitely, and is only valid for TCP connections. It should be noted that a stream may stop before the duration expires if a TCP connection is dropped, or a stop streaming message is received by the instrument.

Requested Data: This field is a bitmask used by the Other Sensors stream only. For other stream channels this value should be left out or set to 0. It is used to set which other sensors will be providing data. The bit values are as follows:

Bit 0 (0x0001) = Temperature + humidity readings
Bit 1 (0x0002) = Heading data (accelerometer + magnetometer readings).
Bit 2 (0x0004) = Time sync data
Bit 3 (0x0008) = Battery data
Bit 4 (0x0010) = Epoch trigger status

3.2 Stop Stream

This command is used to stop **icListen** from streaming data on a specific streaming channel, to the sender of the command. If no data is yet being streamed to the sender of this message, this command will have no effect. This message is used by all binary data streams, and requires no payload.

Byte Offset:	Field:	# Bytes:
0	Type ‘4’ (0x34)	1
1	Sync (0x2A)	1
2	Payload Length	2

Figure 3-2: Stop Message

3.3 Notify

These messages are sent to inform connected users of errors, or stream stoppages.

Byte Offset:	Field:	# Bytes:
0	Type ‘0’ (0x30)	1
1	Sync (0x2A)	1
2	Payload Length	2
4	Notification Code	4

Figure 3-3: Notify Message

Notification Code: This coed indicates the reason the message was sent. Valid values are:

1 = Unable to start stream (no connections available). TCP connections to the port will be dropped after this message is sent

2 = Streaming stopped due to duration timeout. Data will stop streaming after this message, but the TCP connection will remain open.

3 = Stream stopped by stop stream message or “Stop All Streams” command. The TCP connection will be dropped after this message is sent.

4 = Invalid start message received. Data will not be streamed, but the TCP connection will remain open.

5 = Unable to stream data (logging is active). The TCP connection will be dropped after this message is sent.

Firmware Differences:
-HF/AF versions before Release 28 will not drop TCP connections after the “Stream Stopped” message is sent

3.4 Event Header

The event header message is sent up to once per second, before any data is sent for that second. It contains the information necessary for interpreting and synchronising the following data. This message type is only sent from waveform and spectrum data channels.

Byte Offset:	Field:	# Bytes:
0	Type ‘2’ (0x32)	1
1	Sync (0x2A)	1
2	Payload Length	2
4	Event Key Chunk	16
20	Device Info Chunk	k
20+k	Status Chunk	p
20+k+p	Setup Chunk	i
20+k+p+i	Amplitude Scaling Chunk	j

Figure 3-4: Event Header Message

Event Key Chunk: This chunk is used for time keeping and data synchronised.
Device Info: This chunk contains hardware specific information for the instrument.
Status: This chunk contains some extra status information for the instrument.
Setup: This chunk contains the setup used to acquire the following data.
Amplitude Scaling: This chunk contains information used to scale the data to real-world units.

For more detailed information on what is contained in each message chunk, please refer to the appropriate “Message Chunk Details” section.

3.5 Data

This message contains the measured data for the given second.

Byte Offset:	Field:	# Bytes:
0	Type ‘1’ (0x31)	1
1	Sync (0x2A)	1
2	Payload Length	2
4	Event Key	m
4+m	Data	p

Figure 3-5: Data Message

Event Key Chunk: This chunk is used for time keeping and data synchronised. This chunk should match the most recently received event key chunk from the Even Header message. If it does not, the data should be discarded.
Data: This contains acoustic data.

For more detailed information on what is contained in each message chunk, please refer to the appropriate “Message Chunk Details” section.

3.6 Other Sensor Data

This message is sent to give non-acoustic data from the instrument. The chunks returned with this message are configurable using the Requested Data bitmask in the Start Stream message. These messages are sent when new data for the configured sensors is available, and will not necessarily contain all chunks each time it is received.

Byte Offset:	Field:	# Bytes:
0	Type ‘5’ (0x35)	1
1	Sync (0x2A)	1
2	Payload Length	2
4	Temperature/Humidity Chunk	m
4+m	Trigger Status Chunk	p
4+m+p…	Time Sync Chunk	i
	Battery Chunk	
	Heading Chunk	

Figure 3-6: Data Message

Temperature/Humidity Chunk: This chunk contains temperature and humidity readings for the instrument.
Trigger Status Chunk: This chunk contains the current status of the epoch triggers set for the instrument.
Time Sync Chunk: This chunk contains time synchronisation information.
Battery Chunk: This chunk contains battery charge information, and internal voltage measurements.
Heading Chunk: This chunk contains accelerometer and magnetometer readings.

For more detailed information on what is contained in each message chunk, please refer to the appropriate “Message Chunk Details” section.

4 Message Chunk Details

This section describes the format and use of all message chunks used to build the *Event Header* and *Data* messages. The message chunks contained within *Event Header* messages will vary based on the form of data being transmitted. All message chunks use the same basic format:

Byte Offset:	Field:	# Bytes:
0	Chunk Type	1
1	Version	1
2	Chunk Size	2
4	Payload	n

Figure 4-1: Chunk Structure

Chunk Type: This byte identifies the type of chunk being transmitted.

Version: The second character indicates what version of the chunk payload follows. The contents of lower versioned message chunks are preserved in higher versions, so that message parsing software does not need to be updated each time a new version of the message is released. If the chunk version is higher than what the driver was written to parse, the new data fields can be ignored, and the rest of the chunk treated the same as the lower version chunk.

Chunk Size: This is the number of bytes in the payload.

Payload: The payload will vary in length and content, based on the type of chunk being transmitted.

Available chunk types include:

Message	***Code***	***Description***
Event Key	'A' (0x41)	Contains Time/Sequence Number, used to pair data with headers
Data Chunk	'B' (0x42)	Contains Waveform or Spectrum Data
Status	'C' (0x43)	Contains various sensor data, included with data header
Device Info	'D' (0x44)	Contains hardware/firmware details
Wave Setup	'E' (0x45)	Contains the waveform data setup
FFT Setup	'F' (0x46)	Contains the spectrum data setup
Scaling	'G' (0x47)	Contains the details for scaling data counts to real world units
Temp/Humidity	'H' (0x48)	Contains Temperature and Humidity data
Heading	'I' (0x49)	Contains Accelerometer and Magnetometer data
Time Sync	'J' (0x50)	Contains time sync and PPS details
Battery	'K' (0x51)	Contains battery details and status
Trigger Status	'L' (0x52)	Contains the status of all epoch triggers

4.1 Event Key Chunk

The purpose of this chunk is to provide time synchronisation information for data provided by icListen. For waveform and spectrum data channels, this key chunk is also used for matching Data messages with their corresponding Event Header messages. The key chunk present in Data messages, should exactly match the key chunk found in the Event Header message corresponding to that second of data.
Only version 0 of this chunk exists.

Byte Offset:	Field:	# Bytes:
0	Type 'A' (0x41)	1
1	Version (0)	1
2	Chunk Size (12)	2
4	UNIX Time	4
8	Sequence Number	8

Figure 4-2: Event Key Chunk

Unix Time: This is a signed 32bit value UNIX timestamp. UNIX time is defined as the number of seconds since 00:00:00 January 1, 1970 UTC.
Sequence Number: This number corresponds to the number of samples since the current data gathering session within the instrument. It will be reset any time that the time source is synchronised to a new top of second (via a PPS sync signal). This number will count monotonically forward, despite changes to the UNIX time (which can be decode from a PPS signal, or set manually on the web interface or command and control channel). Because of this, it can be used to correct for time changes in data in post-processing if necessary. It will increase by the sample rate each second, and begins with sample 0 at the top of a second.

4.2 Data Chunk

This chunk contains acoustic data, and a small amount of header information to indicate the data format, number of samples, and timing.
Only version 0 of this chunk exists.

Byte Offset:	Field:	# Bytes:
0	Type 'B' (0x42)	1
1	Version (0)	1
2	Chunk Size	2
4	Sample #	4
8	# of channels	1
9	Data Format	1
10	# of Samples	2
12	Data	n
12+n	Padding	k

Figure 4-3: Data Chunk

Sample #: This is the sample within this second of the first sample returned with this message. This can be used to determine if data is missing.
# of Channels: This is the number of data channels represented in this message. All multi-channel data is interlaced. For example, for 2 samples of 3 channel data, data will be ordered as follows: Channel 1/Sample 1, Channel 2/Sample 1, Channel 3/Sample 1, Channel 1/Sample 2, Channel 2/Sample 2, Channel 3/Sample 2.
Data Format: This indicates what format the following data is in. The lower 7 bits of this value indicate the number of bytes per sample, and the highest bit indicates if the data is presented as little endian (1) or big endian (0). The bytes per sample is on a per channel basis (ie: 24bit data will indicate 3 bytes per sample, regardless of whether 1 channel or 4 are present). Valid settings are:

2 = 16 bit Big Endian
3 = 24 bit Big Endian
4 = 32 bit Big Endian (24 bit data packed into 32 bits)
130 = 16 bit Little Endian
131 = 24 bit Little Endian
132 = 32 bit Little Endian (24 bit data packed into 32 bits)

# of Samples: This is the number of samples present in the data field (per channel).
Data: This is the actual acoustic data.
Padding: These bytes are inserted to ensure that the chunk payload is 32bit aligned.

4.3 Status Chunk

This chunk is intended to provide additional status information which is synchronised to the data, and may be useful for monitoring the health of an instrument.
Only version 0 of this chunk exists.

Byte Offset:	Field:	# Bytes:
0	Type 'C' (0x43)	1
1	Version (0)	1
2	Chunk Size	2
4	Temperature	2
6	Humidity	2
8	Battery Percent	1
9	Battery Status	1
10	Data Epoch	1
11	Channel #	1
12	Reserved (0)	12

Figure 4-4: Status Chunk

Temperature: This is internal temperature of **icListen** in tenths of °C (123 = 12.3 °C).
Humidity: This is the internal relative humidity of **icListen** in tenths of % (321 = 32.1 % rh).
Battery Percent: This is the estimated charge % of the internal battery. A value of 0xFF indicates that this data is unavailable.
Battery Status: This code indicates the charging state of the internal battery. Valid codes are:

0 = No Data Available
1 = Battery is Charging
2 = Battery is Discharging
3 = Battery is not Charging or Discharging

Data Epoch: This is a monotonic counter which is incremented each time the ADC must be restarted (occurs when syncing to a new PPS sync signal). The data sequence number will also be set to 0 when this occurs.
Channel #: This the channel number that this status chunk corresponds to.
Reserved: These bytes are reserved for future use.

Firmware Differences:
-HF/AF versions before Release 28 do not update the Data Epoch field

4.4 Device Info Chunk

This chunk provides some basic hardware and firmware information. Version 0 and 1 of this payload exist. Version 0 is used by **icListen** hydrophones, while version 1 is used by the **Host Controller**.

Byte Offset:	Field:	Vresion	# Bytes:
0	Type 'D' (0x44)	All	1
1	Version	All	1
2	Chunk Size	All	2
4	Device Type	All	1
5	Firmware Version	All	3
8	Serial Number	All	2
10	Hydrophone Sensitivity	All	2
12	Channel #	1	1
13	Reserved (0)	1	3

Figure 4-5: Device Chunk

Device Type: This is a code representing the type of device that this unit is. Valid values include:

5 = icListen HF

7 = icListen AF

8 = Host Controller

Firmware Version: This is a null terminated ASCII string which shows the firmware version of the unit (ex: "v1.0.01").

Serial Number: This is the serial number for this device. It is unique to each unit of a given device type.

Hydrophone Sensitivity: This is the sensitivity of the hydrophone in dB re 1µPa (in tenths of dB re 1 µPa).

Channel #: This the channel number that this chunk corresponds to. A channel number of "255" is a flag value used by devices (such as the **Host Controller**) which themselves sdo nor provide a channel of acoustic data.

Reserved: These bytes are reserved for future use.

4.5 Wave Setup Chunk

This chunk provides information on the setup used to collect waveform data.
Only version 0 of this chunk exists.

Byte Offset:	Field:	# Bytes:
0	Type 'E' (0x45)	1
1	Version (0)	1
2	Chunk Size	2
4	Sample Rate	4
8	Gain	2
10	Data Format	1
11	Reserved (0)	1

Figure 4-6: Wave Setup Chunk

Gain: This is the gain applied to waveform data, set in dB. Gain in **icListen HF** is applied digitally.
Sample Rate: This is sample rate at which the data is collected, in samples per second.
Data Format: This indicates what format the following data is in. The lower 7 bits of this value indicate the number of bytes per sample, and the highest bit indicates if the data is presented as little endian (1) or big endian (0). The bytes per sample is on a per channel basis (ie: 24bit data will indicate 3 bytes per sample, regardless of whether 1 channel or 4 are present). Valid settings are:

2 = 16 bit Big Endian
3 = 24 bit Big Endian
4 = 32 bit Big Endian (24 bit data packed into 32 bits)
130 = 16 bit Little Endian
131 = 24 bit Little Endian
132 = 32 bit Little Endian (24 bit data packed into 32 bits)

Reserved: These bytes are reserved for future use.

4.6 Spectrum Setup Chunk

This chunk provides information on the setup used to collect spectrum data.
Only version 0 of this chunk exists.

Byte Offset:	Field:	# Bytes:
0	Type ‘F’ (0x46)	1
1	Version (0)	1
2	Chunk Size	2
4	Sample Rate	4
8	Points Per FFT	2
10	Reference Level	2
12	Processing Type	2
14	Samples Between FFTs	2
16	Accumulation Count	2
18	Weighting Factor	2

Figure 4-7: Spectrum Setup Chunk

Sample Rate: This is sample rate at which the data is collected.
Points Per FFT: This is the number of data points used for each FFT calculation.
Reference Level: This is the reference level that spectrum data results are relative to in dB relative to 1V.
FFT Processing Type: This is a code which determines how FFT data is processed by the instrument. The following processing types are valid:

> **4 (Mean Average):** Avg = sum[Value(0)…Value(Accumulations)]/ Accumulations
> **5 (Peak Detect):** Max value at each frequency taken over the set number of “Accumulations”
> **6 (Exponential Moving Average):** Avg(t) = [(Weight-1) * Avg(t-1) + Value(t)]/Weight

Samples Between FFT’s: This is the number of new data points gathered between each FFT calculation.
FFT’s Accumulated: The number of FFT data sets accumulated together for each spectrum data result.
FFT Weighting Factor: This value is used only when the processing type is set to exponential moving average. It determines the weight on new FFT data sets with respect to the rest of the data in the average by the following formula.

4.7 Amplitude Scaling Chunk

This chunk contains the details for converting the returned values to real world units. There will be one of these chunks for each channel of returned data.
Only version 0 of this chunk exists.

Byte Offset:	Field:	# Bytes:
0	Type ‘G’ (0x47)	1
1	Version (0)	1
2	Chunk Size	2
4	Channel #	1
5	Unit of Measure	1
6	Reserved (0)	1
7	Bytes Per Sample	1
8	Max Count	4
12	Min Count	4
16	Max Amplitude	4
20	Min Amplitude	4

Figure 4-8: Amplitude Scaling Chunk

Channel #: This is the channel that these scaling parameters belong to.
Unit of Measure: This code indicates what type of unit the amplitude values represent.

1 = µV

4 = 2x dB re V (count / 2 * (Max - Min Amplitude) / (Max – Min Count) = dB re V)

Reserved: These bytes are reserved for future use.
Bytes Per Sample: This is the number of bytes per sample in the returned data.
Max/Min Count: The maximum and minimum values returned in the data.
Max/Min Amplitude: The amplitude represented by the full max/min count values, in the units specified in the “Unit of Measure” field.

4.8 Temperature/Humidity Chunk

This is a chunk optionally returned by the other sensor data channel, which contains the internal temperature and humidity readings of a hydrophone.
Only version 0 of this chunk exists.

Byte Offset:	Field:	# Bytes:
0	Type ‘H’ (0x48)	1
1	Version (0)	1
2	Chunk Size	2
4	Temperature	2
6	Humidity	2

Figure 4-9: Temperature/Humidity Chunk

Temperature: This is internal temperature of **icListen** in tenths of °C (123 = 12.3 °C).
Humidity: This is the internal relative humidity of **icListen** in tenths of % (321 = 32.1 % rh).

Firmware Differences:
-HF/AF version 2.1 (Release 28-30) contain an error where the chunk size is given as little endian instead of big endian

4.9 Heading Chunk

This is a chunk optionally returned by the other sensor data channel, which contains the accelerometer and magnetometer readings of a hydrophone.
Only version 0 of this chunk exists.

Byte Offset:	Field:	# Bytes:
0	Type ‘I’ (0x49)	1
1	Version (0)	1
2	Chunk Size	2
4	Accel X	2
6	Accel Y	2
8	Accel Z	2
10	Mag X	2
12	Mag Y	2
14	Mag Z	2

Figure 4-10: Heading Chunk

Acceleration (X,Y,Z): These are the acceleration readings along the x, y, and z axes. These readings are given in milli g-units (1 g-unit = 1 grav). These are signed 16bit values.
Magnetic Field (X,Y,Z): These are the magnetic field readings along the x, y, and z axes. These readings are given in milligauss (mG). Multiplying this value by 100 gives nanotesla (nT) values. These are signed 16bit values.

Firmware Differences:
-HF/AF version 2.1 (Release 28-30) contain an error where the chunk size is given as little endian instead of big endian

4.10 Time Sync Chunk

This is a chunk optionally returned by the other sensor data channel, which contains the time synchronisation data for a hydrophone.
Only version 0 of this chunk exists.

Byte Offset:	Field:	# Bytes:
0	Type ‘J’ (0x4A)	1
1	Version (0)	1
2	Chunk Size	2
4	Sync Status	1
5	System Time Status	1
6	Reserved (0)	2
8	Sync Offset	4
12	PPS Count	4
16	Time Since PPS	4
20	System Time	4

Figure 4-11: Time Sync Chunk

Sync Status: This code indicates is the status of synchronization. Valid codes are:

0 = Sync input/output disabled
1 = Sync output enabled
2 = Sync output enabled (with PPS encoded time)
3 = Sync input enabled (falling edge), no PPS detected
4 = Sync input enabled (falling edge), syncing to detected PPS
5 = Sync input enabled (falling edge), synced to PPS
6 = Sync input enabled (rising edge), no PPS detected
7 = Sync input enabled (rising edge), syncing to detected PPS
8 = Sync input enabled (rising edge), synced to PPS

System Time Status: This code indicates how the system time was set. Valid codes are:

0 = Time value has not been set
1 = Time was manually set
2 = Time set from RTC on reboot
3 = Time was set from file system after reboot
4 = Time was set from NTP
5 = Time was set from GPS
6 = Time was set from PTP
128 = Time set from PPS encoded time (from unset source)
129 = Time set from PPS encoded time (from manually set source)
130 = Time set from PPS encoded time (from source set by RTC on reboot)
131 = Time set from PPS encoded time (from source set by file system on reboot)
132 = Time set from PPS encoded time (from source set by NTP)
133 = Time set from PPS encoded time (from source set by GPS)
134 = Time set from PPS encoded time (from source set by PTP)

Sync Offset: This is the measured time offset in nanoseconds, between the detected PPS edge, and this system’s second edge.

PPS Count: This is a monotonic counter of PPS inputs detected when configured for sync input. This value will be set to 0 if sync input is not enabled.

Time Since PPS: This is the number of seconds that have elapsed since the last PPS edge was detected.

System Time: This is the UNIX time value of **icListen**’s system clock. UNIX time is defined as the number of seconds since 00:00:00 January 1, 1970 UTC.

Firmware Differences:
-HF/AF version 2.1 (Release 28-30) contain an error where the chunk size is given as little endian instead of big endian

4.11 Battery Chunk

This is a chunk optionally returned by the other sensor data channel, which contains the battery and internal power information for a hydrophone.
Only version 0 of this chunk exists.

Byte Offset:	Field:	# Bytes:
0	Type 'K' (0x4B)	1
1	Version (0)	1
2	Chunk Size	2
4	Charging Status	1
5	Charge Percent	1
6	Charge (mAh)	2
8	Capacity	2
10	Battery Current	2
12	Battery Voltage	2
14	Battery Temperature	2
16	Board Voltage	2
18	Microprocessor Voltage	2

Figure 4-12: Battery Chunk

Charging Status: This is a code which indicates the charging state of the battery. Valid codes are:

0 = No Data Available
1 = Battery is Charging
2 = Battery is Discharging
3 = Battery is not Charging or discharging

Charge Percent: This is the estimated percentage of full capacity that the battery is charged to.
Charge: This is the estimated charge currently stored in the battery, measured in mAh.
Capacity: This is the capacity of the battery when fully charged, measured in mAh.
Battery Current: This is current flowing into the battery, measured in mA. A negative current indicates that current is flowing out of the battery (discharging).
Battery Voltage: This is the voltage measured on the battery terminals, measured in mV.
Battery Temperature: This is temperature of the battery, measured in tenths of °C (303 = 30.3 °C).
Board Voltage: This is regulated supply voltage to the **icListen**, measured in mV.
Microprocessor Voltage: This is the supply voltage to the main microprocessor, measured in mV.

The battery state can report a status code of 3 (NOT charging or discharging), for one of 3 reasons: the battery may be fully charged, the status may have been updated during the switch between charging and discharging, or the charging may have been stopped due to high battery temperature.

Firmware Differences:
-HF/AF version 2.1 (Release 28-30) contain an error where the chunk size is given as little endian instead of big endian

4.12 Trigger Status Chunk

This is a chunk optionally returned by the other sensor data channel, which contains the epoch trigger status for a hydrophone.
Only version 0 of this chunk exists.

Byte Offset:	Field:	# Bytes:
0	Type ‘L’ (0x4C)	1
1	Version (0)	1
2	Chunk Size	2
4	# Triggers	1
5	Epoch 1 Status	1
6	Epoch 2 Status	1
7	Epoch 3 Status	1
8	Epoch 4 Status	1
9	Epoch 5 Status	1
10	Reserved (0)	2

Figure 4-13: Trigger Status Chunk

# Triggers: This is the estimated percentage of full capacity that the battery is charged to.
Epoch *n* Status: This code indicates the state of trigger detection/assertion. Valid codes are:

0 = No Signal, Event Inactive (or trigger disabled)
1 = Signal Present, Event Inactive
2 = Signal Present, Event Active
3 = No Signal, Event Active

Reserved: These bytes are reserved for future use.

An epoch Event, is the devices reaction to s signal being detected. This event can be a transmitted message over the epoch message stream, logging that message to a file, or logging acoustic data (waveform and/or spectrum files).

The following figure shows a trigger going through all stages as a signal appears and disappears. It begins in state 0: there is no signal and the event is inactive. When the desired signal appears, it enters state 1: the signal is present, but the event is still inactive as the signal has not been present for long enough. After the signal has been present for long enough to cause an event trigger, the status changes to 2: the signal present and the vent active. Once the signal disappears, the status changes to 3: there is no signal, but the event is still active. This status lasts for the duration of the configured event hold and then the status returns to 0.
A status of 0 is also set when no trigger is configured.

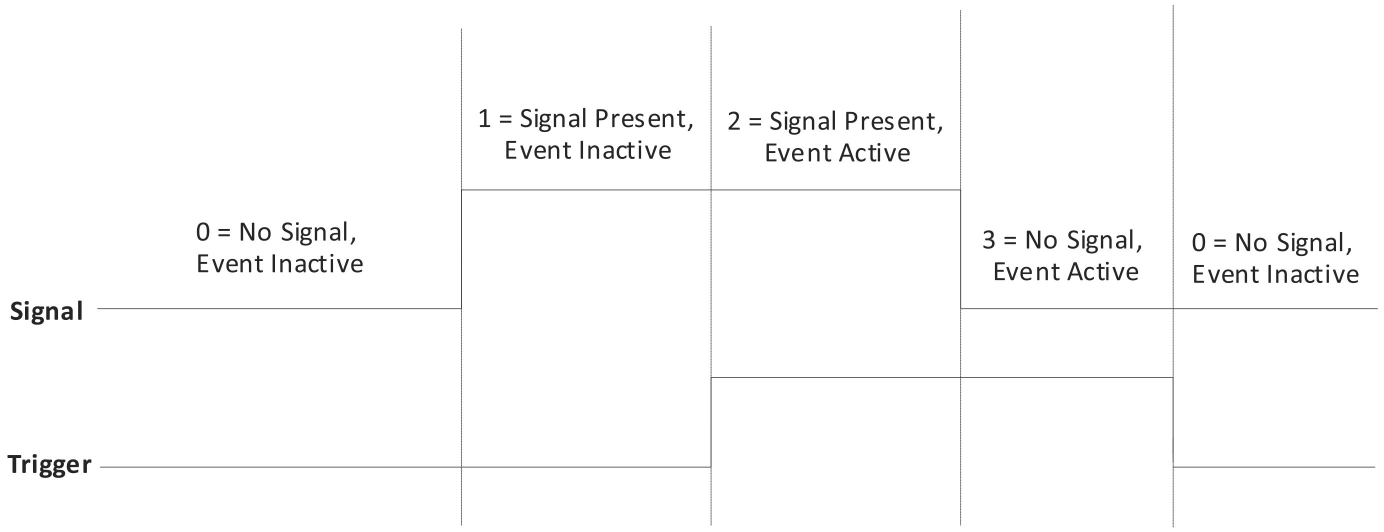
1 = Signal Present, Event Inactive
2 = Signal Present, Event Active
0 = No Signal, Event Inactive
3 = No Signal, Event Active
0 = No Signal, Event Inactive
Signal
Trigger

Figure 4-14: Trigger Status Example

Firmware Differences:
-HF/AF version 2.1 (Release 28-30) contain an error where the chunk size is given as little endian instead of big endian