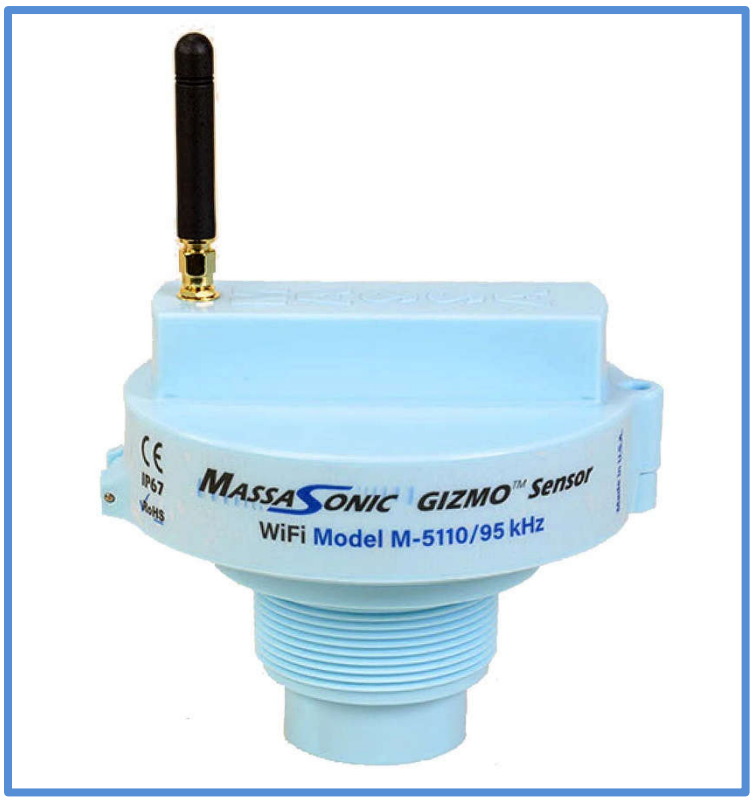
MASSA
GENERATIONS AHEAD IN SONAR & ULTRASONIC TECHNOLOGY

MASSA PRODUCTS CORPORATION
280 LINCOLN STREET, HINGHAM, MA 02043-1796 U.S.A.
781-749-4800 FAX: 781-740-2045 www.massa.com

MASSASONIC® Gizmo®
Wireless Ultrasonic Level Sensor
Advanced Users Programming Guide



June 6, 2024 rev 1.0

Table of Contents

1 Introduction

This guide contains detailed information for Gizmo®'s Ultrasonic Sensor WiFi communications and follows the Gizmo® Installation and Getting Started Guide. For additional information including data sheets, application software and application notes, please refer to Massa website:

www.massa.com/industrial/ultrasonic-sensors/gizmo/ (currently not available)

Notes:

1. MassaSonic Multi-Sensor Application (MMSA) is Gizmo's support software with user-friendly GUI interface and software drivers that take care of all detailed communication. See Section 5 for an MMSA overview and the **MMSA Software Guide** for detailed use.
2. Sections 3 and 4, contain JSON scripts that are required if developing communication software. Section 6 Communication Specification is targeted toward technical end users developing applications to communicate with Gizmo. The JSON scripts of sections 3 and 4 and section 6 can be skipped when using Massa's MMSA software for Gizmo communication.
3. When developing communication software and writing JSON scripts of sections 3, 4, and 6:
   - When using MQTT publish the JSON object to the MQTT topic specified. The "Action":"verb" label value pair is ignored.
   - When using Local Host the "Action":"verb" label value pair is required and the MQTT topic is not used.

2 Overview

The MassaSonic® Gizmo® Ultrasonic Sensors combine Massa's 75-plus years of experience in electroacoustics with state-of-the-art analog and microprocessor hardware and software design. The result is the most versatile, easiest-to-use ultrasonic sensor on the market. The Gizmo Family of Sensors consists of sensors that operate at different frequencies which determine sensing ranges in a typical tank level application.

In operation, Gizmo® Sensors generate a high-frequency ultrasonic pulse, then measures the time it takes for the reflected echo to return from a target and then calculates the target distance using the speed of sound. The value of the speed of sound, which is a function of temperature, is determined by the sensor using its internal temperature probe. The status data produced is transported via WiFi or Cellular to the system Host.

Gizmo's communication options of WiFi or Cellular wireless technology, combined with MQTT or Local Host network protocols are designed to meet the needs of a very large variety of applications and installations.

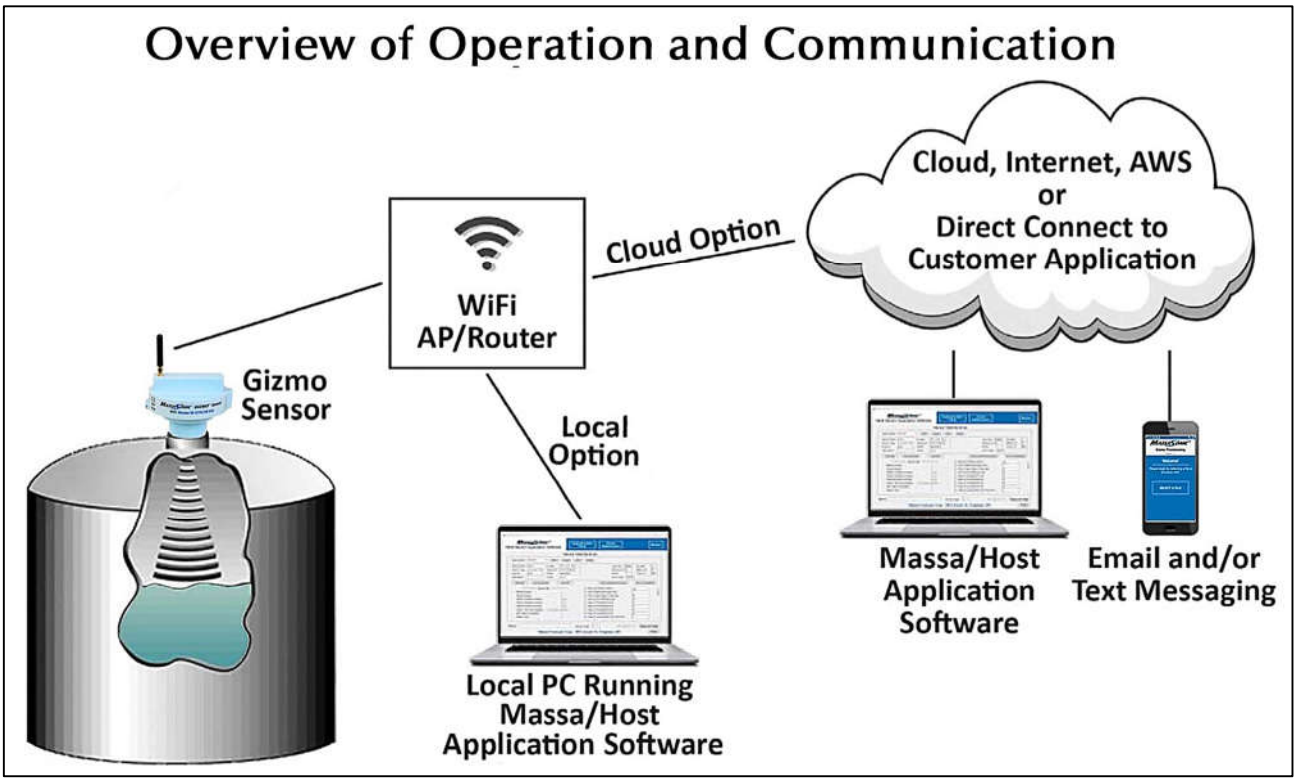
Gizmo provides a complete echo system. Gizmo sensor in combination with MMSA software provide all the tools to communicate, configure sensor, monitor measurements, save histories, and analyze data.

2 Overview (continued from previous page)

Gizmo Communications

Gizmo features wireless access functionality to connect to preexisting in-plant WiFi network installations or to commercial Cellular carriers.

- Gizmo utilizes both the industry standard MQTT transport and the streamlined Local Host (tunneling) transport, for secure JSON (JavaScript Object Notation) data interchange.
- Gizmo WiFi securely connects to a WiFi network and then to an MQTT Broker or Host. The MQTT Broker can be in the cloud or hosted on-site.
- The MQTT Broker is the gateway to all Gizmos. Users will access Gizmo event data, configuration, and waveform diagnostics through the MQTT Broker.
- Gizmo's Local Host protocol, a simpler JSON transport connecting directly to MMSA or user proprietary software without the need for an MQTT broker and overhead.
- Gizmo is supported by MassaSonic® Multi-Sensor Application (MMSA), a software package which provides an easy-to-use GUI interface to Gizmo data via its built in MQTT and Local Host drivers.
- Gizmo provisioning sets WiFi details, broker connection, and security settings required for network and broker use. Use either MMSA, cloud based automated Fleet Provisioning, or Gizmo's Provisioning Smartphone App.



2 Overview (continued from previous page)

User Selectable Wireless Technology and Transport Protocols

Gizmo offers the option of both WiFi and Cellular wireless technologies and the flexibility of MQTT or Local Host transport protocols.

Choose wireless based on sensor location, network connectivity, wireless service, and public or private access. Choose what works best for your data use, monitoring requirements, security requirements. Configure your Gizmo network:

- Configure using Local Host and WiFi for a more streamlined, enclosed, private network with system access via the local MMSA host.
  Or
- Configure your system to use MQTT with both WiFi and Cellular networks for more advanced and flexible cloud-based service. MQTT is an industry standard supporting brokered distribution, database recording, multiple access clients, alarm notifications, and secure web site access with current and historical data.
  Or
- Mix and match both wireless technologies and transport protocols, from one Gizmo to the next, building best case installations no matter your location. Regardless of technology and protocol combinations, all deliver the same compatible Gizmo JSON data payload.

Note: Wireless WiFi and Cellular options are factory ordered. Transport protocols are selected during Gizmo provisioning.

Gizmo Data Security

All MQTT and Local Host communication is secure. Gizmo implements TLS-based mutual authentication using X.509 certificates to protect and encrypt data in transit.

Gizmo data communication accepts security from a variety of sources. Certificates may be customer purchased from a Trusted Certificate Authority, provided by AWS during the provisioning process, or generated as Self Signed Certificates produced by MMSA.

Certificates and provisioning data are stored in Gizmo's internal non-volatile and non-accessible memory.

Gizmo with MQTT / JSON Transport

Gizmo WiFi and Cellular models both support MQTT transport with JSON data payload. MQTT with JSON data interchange is a lightweight protocol design for limited resource embedded systems. This implementation of Gizmo communications is aimed at customers using (or soon to be using) the popular, industry standard MQTT broker with JSON data payload. See section 6.1 MQTT Data Transport for details.

MQTT/JSON communication is selected by default during Gizmo provisioning.

Note: With an MQTT broker, multiple clients, including MMSA, can communicate with Gizmo. Multiple user custom MQTT client and multiple instances of MMSA can subscribe to Gizmo topics, monitor Gizmo events, and publish new settings to the same Gizmo.

Gizmo with Local Host / JSON Transport

Gizmo WiFi and Cellular models also support Local Host data transport. Local Host is an alternative to MQTT, transporting the same JSON data payload directly via the underlying TCI/IP protocol or tunneling. This eliminates the MQTT protocol layer, MQTT topic overhead, and the required MQTT broker, simplifying messaging and required support services.

This Local Host implementation is aimed at more streamlined systems with a local network or cloud-based host running a single instance of MMSA or user proprietary software. Local Host is selected during Gizmo provisioning.

See section 6.2 Local Host Data Transport for details.

2 Overview (continued from previous page)

Gizmo WiFi Communication Profiles

Gizmo stores network login and security information in two profiles, the Primary profile, and the Fallback profile. Each profile includes a complete set of WiFi and Broker/Host information necessary to connect and transport data.

On wakeup, Gizmo first makes 3 attempts to connect using the Primary profile. If unsuccessful, Gizmo then makes 3 attempts using the Fallback profile. If still unsuccessful, Gizmo goes back to sleep until its next wakeup time.

Real Time Clock

Gizmo includes a Real Time Clock (RTC) to maintain accurate time to timestamp events. The RTC is checked on each Gizmo wakeup against an internet NTP time server. If the RTC time does not match NTP time, Gizmo converts NPT time to its internal J2000 epoch time format and saves it in the RTC. This timestamp is recorded with each Measurement Event.

3 Gizmo Operating Modes

Gizmo Operating Modes

Gizmo includes seven operating modes tailoring operation to application requirements and battery life. Six modes support battery power with the seventh mode requiring line power.

Here is a list and brief description of operating modes. It will be followed with additional details.

- **Report On Wake** is the default operation and sends status (tank level) upon waking up based on the programmed Sleep Time setting.
- **Report On Delayed Event Count** is a mode that sends status reports as a block minimizing radio traffic and saving battery power. This mode features unscheduled reporting if over and under Alarms are enabled.
- **Wake & Catch Up** occurs automatically when WiFi or broker connections failed upon previous attempts upon waking up. When Gizmo wakes up and finally connects to the WiFi network, it will automatically send a block of records that were not previously sent.
- **Tank Fill / Empty Override** provides a fast-sampling rate during tank fill or empty. This temporary override of the set Sleep Rate occurs at a programmed scheduled time and includes a end time to halt this operation. See below for additional information about this mode.
- **Fast Sampling Override** is a mode that provides a temporary burst of fast sampling for fast changing tank levels upon next scheduled awake for a fixed length of time. See below for additional information about this mode.

Future Operation

- ***Report On Request From Host*** *is a mode where sensor performs its level measurement upon expiration of Sleep Time but provides the status report only upon request by the host. Gizmo will timeout if host does not request for data or other commands like changing settings.*
- ***Always Powered*** *is a mode that keeps the sensor on the WiFi network. Gizmo will continue to report tank level at programmed sleep rate, report tank level upon request, change sensor settings, update Gizmo clock, provide historical records of past tank levels, provide diagnostic waveforms, or perform over the air firmware updates.*

3 Gizmo Operating Modes (continued from previous page)

Gizmo Operating Mode – Report on Wake (default mode)

Here is a detailed list for this operating mode:

- Gizmo wakes periodically when "Sleep Time" expires.
- Makes a range measurement.
- Saves this measurement Event in a non-volatile history record.
- Powers the radio, connects to the WiFi network, to the MQTT broker, and subscribes to data requests, data writes, and commands.
- If Gizmo does Not successfully connect with the broker, Gizmo goes back to sleep without reporting this Event
- If Gizmo does connect, it publishes this measurement Event (Type "Report on Wake"). See report details in Table 1.
- Once connected, Gizmo will stay awake to read and process available, subscribed data from the broker.
- When broker data is idle for 3 seconds (programmable), Gizmo will break the network connection and go to sleep.

Set this mode by sending:

{"Action":"Write", "OpMode": 0} //MQTT Publish Topic …/write/config

Continued next page

3 Gizmo Operating Modes (continued from previous page)

Gizmo Operating Mode – Report on Delayed Event Count

This operating mode minimizes radio traffic thus saving battery power. Here is a detailed list for this operating mode:

- Gizmo wakes periodically when "Sleep Time" expires.
- Makes a range measurement.
- Saves this measurement Event in non-volatile history record.
- If the number of history records stored since the last report does not exceed the "Delayed Report Count" Gizmo goes back to sleep.
- If the number of history records stored since the last report is greater than or equal to the "Delayed Report Count" Gizmo:
  - Powers the radio, connects to the WiFi network, to the MQTT broker, and subscribes to data requests, data writes, and commands.
  - Publishes all Events recorded since the last successful report. (Type "Report on Delayed Count"). See report details in Table 1.
- If the measured range just collected during this wake is > "High Alarm" or < "Low Alarm" Gizmo immediately:
  - Powers the radio, connects to the WiFi network, to the MQTT broker, and subscribes to data requests, data writes, and commands.
  - Publishes all Events recorded since the last successful report. (Type "Report on Alarm"). See report details in Table 1.
- Once connected, Gizmo will stay awake to read and process available, subscribed data from the broker.
- When broker data is idle for 3 seconds (programmable), Gizmo will break the network connection and go to sleep.

Set this mode by sending:

{"Action":"Write", "OpMode": 1} //MQTT Publish Topic …/write/config

Gizmo Operating Mode – Wake and Catch Up

This operating mode reports previously not reported events due to WiFi or broker connection failures. Here is a detailed list for this operating mode:

- Gizmo wakes periodically (every "Sleep Time" seconds).
- Makes a range measurement.
- Saves this measurement Event in a non-volatile history record.
- Powers the radio, connects to the WiFi network, to the MQTT broker, and subscribes to data requests, data writes, and commands.
- If Gizmo Does Not successfully connect with the broker, Gizmo goes back to sleep without reporting Events.
- If Gizmo Does connect, it publishes all Events recorded since the last successful report. (Type "Report Catchup"). See report details in Table 1.
- Once connected, Gizmo will stay awake to read and process available, subscribed data from the broker.
- When broker data is idle for 3 seconds (programmable), Gizmo will break the network connection and go to sleep.

Set this mode by sending:

{"Action":"Write", "OpMode": 2} //MQTT Publish Topic …/write/config

3 Gizmo Operating Modes (continued from previous page)

Gizmo Operating Mode – Tank Fill / Empty Override

The Tank Fill / Empty Override operating mode provides for fast sampling during tank fill or empty operations. Here is a detailed list for this operating mode:

- Gizmo is set to Fill Mode by MMSA or by sending the JSON FillEmpty object during a normal wakeup event. Values set are the Fill or Empty Override Mode, the override target level, the override Sample Rate (overrides original sleep time), the override Start Time, and the override maximum time. After override instructions are sent, Gizmo goes back to sleep.
- When Gizmo wakes up at the override Start Time it changes Sleep Time to the override Sample Rate, measures level, and goes back to sleep.
- Gizmo continues to wake up and measure at the override Sample Rate.
- If measured level has not reached the override Target, alarm condition, or timeout, Gizmo goes back to sleep.
- If measured level reaches the override Target
  - Powers the radio, connects to the WiFi network, to the MQTT broker, and subscribes to data requests, data writes, and commands.
  - Publishes a Measurement Event Report with the range achieved and type "Override Done". See report details in Table 1.
  - Gizmo then terminates the override mode and restores normal sleep time.
- If the elapsed time of the override mode reaches the timeout
  - Powers the radio, connects to the WiFi network, to the MQTT broker, and subscribes to data requests, data writes, and commands.
  - Publishes a Measurement Event Report with the current range and type "Override Timeout". See report details in Table 1.
  - Gizmo then terminates the override mode and restores normal sleep time.
- Once connected, Gizmo will stay awake to read and process available, subscribed data from the broker.
- When broker data is idle for 3 seconds (programmable), Gizmo will break the network connection and go to sleep.

Set this temporary override mode by sending:

```
{"Action":"Write",
  "FillEmpty": {
  "StartTime": 697123887,          //J2000 epoch time
  "OverrideMode": 1,               //Override mode is 1 for Fill, 2 for Empty
  "OverrideTarget": 57.3,
  "OverrideRate": 25,
  "OverrideTimeout": 600
  }}}                              //MQTT Publish Topic …/write/config
```

3 Gizmo Operating Modes (continued from previous page)

Gizmo Operating Mode – Fast Sample Override

The Fast Sample mode provides a temporary burst of fast sampling when tank levels may be changing quickly. Here is a detailed list for this operating mode:

- Gizmo is set to Fast Sample mode by MMSA or by sending the JSON FillEmpty object during a normal wakeup event. Values set are the Fast Sample Rate (overrides original sleep time), the Fast Sample Start Time, and the Fast Sample Maximum Time. Gizmo goes back to sleep.
- When Gizmo wakes up at the Fast Sample Start Time, it changes Sleep Time to the Fast Sample Rate and starts the Fast Sample override mode.
- Gizmo will continue to take a range sample and report results at the Fast Sample Rate.
- If Fast Sample Rate is greater or equal to 30 seconds:
  - At sample time Gizmo will power the radio, connect to the WiFi network, to the MQTT broker, and publishes a Measurement Event Report (Type "Report on Fast Sample"). See report details in Table 1.
  - Gizmo will then go back to sleep.
- If Fast Sample Rate is less than 30 seconds
  - Gizmo does not sleep and leaves the radio powered and connected to shorten report times.
  - At sample time, Gizmo will publish a Measurement Event Report (Type "Report on Fast Sample"). See report details in Table 1.
- When the Fast Sample Maximum Time expires:
  - Gizmo will terminate the override mode and restores normal sleep time.

Set this temporary override mode by sending:

```
{"Action":"Write",
  {"StartTime": 697123887,                    //J2000 epoch time
  "OverrideMode": 3,
  "OverrideRate": 25,
  "OverrideTimeout": 600
  }}                                          //MQTT Publish Topic …/write/config
```

3 Gizmo Operating Modes (continued from previous page)

Gizmo Operating Mode – Report On Request (Future feature)

When firmware is released, this feature will allow additional commands to be requested by the host before sensor goes back to sleep. Here is a detailed list for this operating mode when it becomes available:

- Gizmo wakes periodically (every "Sleep Time" seconds).
- Makes a range measurement.
- Saves the measurement in non-volatile History record.
- Powers the radio, connects to the WiFi network, to the Host, and waits for a host request.
- In this mode once Gizmo is connected, the Host is in control of what data to read or write. The Host will typically request Gizmo status, which includes the reason for waking up. From that information, the Host may request the most recent reading or recent alarm condition. The Host may also request waveform data or read and write operating parameters.
- Gizmo will stay awake as long as the host continues communication.
- If the Host communication is idle for more than 3 seconds (programmable), Gizmo will break the network connection and go to sleep.

Gizmo Operating Mode – Stay Always Powered (Future feature)

When firmware is released, this feature will keep the WiFi radio powered allowing access to sensor at all times allowing any commands to be issued and executed. This is useful for setting up sensor, perform testing and obtaining diagnostic waveforms, and constant level sensing for applications that may need such a feature without concerns for battery power. Here is a detailed list for this operating mode when it becomes available:

- When Gizmo powers up, it immediately powers the radio, connects to the WiFi network, and connects to the Host.
- At this point, Gizmo is in an idle mode waiting for Host direction. Gizmo does not sleep or disconnect in this mode.

4 Gizmo Programming Features

Ultrasonic Range Measurement

Range to target (water level) is measured by driving the transducer with a tone burst, referred to as a ping, and waiting for the arrival of a returned echo for the target. The returning echo is captured when the received signal level goes higher than the programmed threshold. This echo capture event is recorded with the precise elapsed time from the start of ping until echo capture. Using the medium's speed of sound, the range to the target can be calculated from the echo's arrival time.

The acoustic environment will vary with installation. Factors such as range, medium, temperature, and obstacles in the acoustic path will affect sound propagation. Gizmo implements the following set of features to accurately determine the distance to targets in diverse and varying environments.

The following ping parameters are programmable to adjust for acoustic environment with optimized battery life.

Time Varying Threshold

The expected signal level of the returning echo decreases as distance to the target surface increases. To compensate for this, Gizmo implements a programmable 4 step time varying threshold. The value for each threshold step and the time of each threshold step change may be programmed based on the specific acoustic environment.

Threshold Values are programmed as a percentage of the maximum echo level. Threshold Switch Times are programmed in inches from the start of ping.

Ping Transmit Cycles

To boost weak echoes, the user may program the number of cycles in the ping tone burst. More ping cycles means more acoustic energy transmitted, which translates to higher-level echo signals from distant targets. The number of transmit cycles is programable for each ping.

Ping Transmit Power

Gizmo employs two power levels to drive the transmit ping. For close targets, Gizmo would use low power, which is adequate for close target capture and conserve battery power. For more distant targets, Gizmo will use high power, which uses more battery power but gets the target. Transmit power level, set as Low or High, and is programmable with each ping.

Time Switched Gain

To further boost weak signals, Gizmo implements a Time Switched Gain increase which amplifies a weak acoustic signal. Initially, all pings start with low gain until the gain switch time is reached. At gain switch time, high gain is enabled. The switch time is programmed in inches for each ping.

Blanking Range

When Gizmo pings, the transmitted acoustic energy takes some time die out. To guard against this energy being detected as a false target, ping Blank Range prevents target capture for the specified blank range time. Blank Range is programmed in inches for each ping.

End of Range

End of Range, programmed in inches, terminates the ping process at the specified time. In the case of up-close targets, terminating the ping process after expected target capture will conserve battery life. End of Range is programmed for each ping.

4 Gizmo Programming Features (continued from previous page)

Ping Sequence Strategy

Gizmo uses a set of pings with varying tone bursts and transmit power to capture the echo. Gizmo's default ping settings are selected to conserve battery and assumes the target is at close range. The ping sequence starts with a 1 cycle ping (minimum transmit energy) and low power. When an echo is captured, the ping sequence terminates, and range is reported. If no echo is captured, Gizmo continues with the ping sequence listed below until an echo is captured or the ping sequence is complete.

1. 1 cycle short ping with low power
2. 10 cycle long ping with low power
3. 1 cycle short ping with high power
4. 10 cycle long ping with high power

If after all pings are complete and no echo is captured, the No Echo state is reported.

If an echo is captured, range is calculated from the captured time of flight (time from ping start to echo capture) and the medium's speed of sound (determined by sensor's temperature probe).

Settings Ping Control

The following is an example of a JSON message that will set all ping control parameters for one ping. Note that Gizmo supports four pings.

```
{"Action":"Write",                    //MQTT Publish Topic …/write/config
  "PingID3": {
  "TxCycles_3": 8,
  "TxHiPwrEnable_3": 1,
  "BlankRange_3": 2.5,
  "GainSwitch_3": 40.6,
  "EndOfRange_3": 144.0,
  "Threshold_3A": [85, 75, 50, 40],
  "ThreshDistance_3A": [5.50, 10.7, 15.0]
}}
```

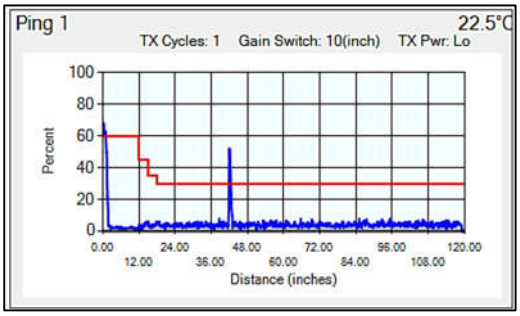
Acoustic Waveform Diagnostics

To aid installation in complex acoustic environments, Gizmo includes a diagnostic tool that returns the received acoustic signal waveform data for the requested single ping or for all four pings. This signal will not only show the target and signal strength relative to thresholds but also other possible interfering objects in the acoustic path. Using this analysis, appropriate thresholds, ping sequencing, gain, and blanking range can be programmed to optimize target capture and range accuracy.

The following is an example of a JSON message that will request a waveform using ping 2. Pings are numbered 1 - 4. Use 5 to retrieve all 4 ping waveforms with one command.

```
{"Action":"Request",                  //MQTT Publish Topic …/request/waveformping
    "WaveformPing": {
    "WFPing": 2,
}}
```

Example of a single waveform as seen in MMSA Software



4 Gizmo Programming Features (continued from previous page)

Event Reports

When Gizmo wakes in either the Report on Wake or Wake and Catch Up modes, Gizmo reports the following JSON formatted measurement event. See Table 1 for parameter details.

{"GizmoID":"5C027209A1E0","GizmoVersion":"1.21","Action":"Response",
"Event":{"Type": 0,"EventTime":686951605,"Range":1.25,"Temp":23.7,
"Volts": 3.4,"RSSI":-67,"PingUsed":3,"Status":0,"SignalStrength":75,
"LastEventIndex":238
}}

When Gizmo wakes and has more than one event to report (Report on Delayed Event Count or Wake and Catch Up modes), Gizmo will report an array of events in a single packet as follows.
Notes:

1. If Gizmo has more than eight events to report it will send multiple packets with each packet containing up to 8 events.
2. Multi-Event reports also include the total Events reported in the first packet
3. Multi-Event reports are limited to a maximum of 50 most recent events.

{"GizmoID":"5c027209a1e6","SensorFmwrVer":"0.28","Action":"Response","TotalEvents":6,
"Event":
[
{"Type":2,"EventTime":730415172,"Range": 51.66,"Temp": 24.00,"Volts": 3.50,"RSSI":61,
"PingUsed":1,"Status":"0x01","SigStrength":40,"LastEventIndex":761},
{"Type":2,"EventTime":730415226,"Range": 0.00,"Temp": 24.00,"Volts":3.55,"RSSI":61,
"PingUsed":4,"Status":"0x03","SigStrength":0,"LastEventIndex":762},
{"Type":2,"EventTime":730415273,"Range": 51.79,"Temp": 24.00,"Volts":3.53,"RSSI":61,
"PingUsed":1,"Status":"0x01","SigStrength":36,"LastEventIndex":763},
{"Type":2,"EventTime":730415296,"Range": 51.71,"Temp": 23.50,"Volts":3.51,"RSSI": 61,
"PingUsed":1,"Status":"0x01","SigStrength":36,"LastEventIndex":764},
{"Type":2,"EventTime":730415310,"Range": 51.73,"Temp": 24.00,"Volts":3.54,"RSSI":61,
"PingUsed":1,"Status":"0x01","SigStrength":36,"LastEventIndex":765},
{"Type":2,"EventTime":730415338,"Range": 51.70,"Temp": 23.50,"Volts":3.50,"RSSI":61,
"PingUsed":1,"Status":"0x01","SigStrength":36,"LastEventIndex":766}
]}

Event History

Gizmo records up to 1000 Measurement Events, as described in Table 1, in its internal Event Log. When 1000 events have been recorded, the next event overwrites the oldest recorded event.

The Event Log is available to read back by user software. Read the event log by specifying the start event index (1 true 1000) and the number of Events to read. Note the index of the most recent or last recorded Event is reported in each Event report. Request history with the number of event records and the starting event index as shown below.

{ "Action":"Request", //MQTT Publish Topic …/request/history
 {
 "RequestCount": 2,
 "StartIndex": 51
 }
}

Event History can be cleared with:

{"Action":"Command", //MQTT Publish Topic …/command/direct
 "ClearHistory":{}
}

4 Gizmo Programming Features (continued from previous page)

FOTA (Firmware Over-the-Air) Update

Gizmo includes a FOTA capability that provides remote updates of both sensor and radio firmware. Under user control, sensor or radio firmware is downloaded and installed from Massa's GitHub repository.

Massa will notify users when new firmware versions are available. The user, at a time of their choosing, will send one of the following commands to update firmware.

When Gizmo is awake send:

```
{"Action":"Command",                    //MQTT Publish Topic …/command/direct
  "SensorFOTA":{}}
```

for Sensor firmware update

or

```
{"Action":"Command",                    //MQTT Publish Topic …/command/direct
  "RadioFOTA":{}}
```

for Radio firmware update

When Gizmo receives the command, it will check the repository revision. If newer than Gizmo's current revision, Gizmo will proceed to download and install the new revision.

5 Gizmo Support Software – MMSA Overview

MassaSonic Multi-Senor Application (MMSA) is Massa's Gizmo support software for communicating with Gizmo. MMSA software is a combination of the Site Host application integrated with the Gizmo's Sensor Maintenance tool set.

The Site Host is a user-friendly graphic interface (GUI) presenting high-level monitor view and site setup features for tank installation. The Site Host is the user side of MMSA and includes views of tank inventory, a tank level history log, tank status, and inventory alarms. For demonstration purposes, the Site Host implements a tank farm application.

The Sensor Maintenance tool set is the underlying technical interface to sensors, connecting and transporting remote Gizmo data. The tool set takes care of the complexities of multi-channel communications, protocol translation, messaging queues/timing, and sensor maintenance. The tool set is designed to support wired and wireless sensors connected directly to the host platform, connected via a local WiFi router, or cloud-based sensors connected via the internet.

MMSA includes built-in capability for both MQTT and Local Host transport.

NOTE: Using MMSA eliminates the need to write software to implement the Gizmo communication specifications of section 6.

See the **MassaSonic Multi-Sensor Application (MMSA) Software Guide** for detailed operation and use.

6 Gizmo Communication Specifications

This section details Gizmo's communication protocols and messaging format.

Gizmo can be provisioned to use either the industry standard MQTT (Message Queuing Telemetry Transport) or Massa's Local Host transport (formerly referred to as TCP Direct or Tunneling).

- With **MQTT transport** Gizmo communicates through, a usually cloud based MQTT broker. The broker distributes messages to one or more authorized hosts (MMSA or customer proprietary software).
- With **Local Host transport**, Gizmo communicates using a local network with MMSA Host software running on a local computer.

In either transport case Gizmo's data interchange is encoded as a JSON (JavaScript Object Notation) message payload.

This section is targeted toward technical end users developing applications to communicate with Gizmo. When developing communication software and writing JSON scripts of sections 3, 4, and 6 note the following:

- When using MQTT publish the JSON object to the MQTT topic specified. The "Action":"verb" label value pair is ignored.
- When using Local Host the "Action":"verb" label value pair is required and the MQTT topic is not used.

6.1 MQTT Data Transport

When provisioned for MQTT, Gizmo reports data and receives commands through MQTT transport. MQTT is a brokered messaging protocol using a publish/subscribe model. Messages are sent to the broker with an accompanying topic. Published messages are distributed by the broker to all devices subscribed to the published topic. Gizmo messages are JSON (JavaScript Object Notation) formatted packets.

Gizmo reports data and responds to commands by publishing to their corresponding topics and receives commands through subscriptions to their corresponding topics. Topics include qualifier fields that are set during Gizmo provisioning. Topic qualifiers allow the user to group and differentiate topics associated with different Gizmo locations, usage, or applications. Gizmo's reference implementation uses the Amazon AWS IoT Core MQTT Broker. See section 7 Gizmo Provisioning Overview for an overview.

All MQTT communication is secure. Gizmo implements TLS-based mutual authentication using X.509 certificates to protect and encrypt data in transit.

MQTT Topics

MQTT Topics Are in the Form: Owner/Group/Device ID/Task/Data Set where the forward-slash ("/") separates the different topic parameters.

Parameters:

Owner	Is user-assigned that provides topic filtering. Management software can subscribe to a particular owner.
Group	Group is a user-assigned that provides for topic filtering. Management software can subscribe to a particular group or subscribe to all groups with a wild card.
Gizmo ID	Gizmo ID is the sensor's unique identifier. Default IDs are either sensor WiFi MAC address or Cellular IME. Gizmo ID may be changed by the end-user during provisioning. The Gizmo ID is typically left blank during provisioning and will automatically default to either the radio's MAC or IMEI, depending on the radio technology.

6 Gizmo Communication Specifications (continued from previous page)

6.1 MQTT Data Transport (continued from previous page)

Task	Task is the topic's function: Report, Request, Response, Write, Command
Data Set	Is the specific set of data transacted. The Data Sets are Event, Configuration, Waveform, History, and Command

Subscription topic filtering can make use of Owner, Group, Gizmo ID, and/or Task.

MQTT Publish Topics:

Report an event	owner/gizmo_g1/f8300237c944/report/event
Respond with configuration data	owner/gizmo_g1/f8300237c944/response/config
Respond with waveform data	owner/gizmo_g1/f8300237c944/response/waveform
Respond with history data	owner/gizmo_g1/f8300237c944/response/history

Gizmo Subscribes to the Following Wildcard Topic Categories:

Request	owner/gizmo_g1/f8300237c944/request
Write	owner/gizmo_g1/f8300237c944/write
Command	owner/gizmo_g1/f8300237c944/command
Provision	owner/gizmo_g1/f8300237c944/provision

Gizmo Processes the Following Specific Topics:

Request configuration	owner/gizmo_g1/f8300237c944/request/config
Request waveform	owner/gizmo_g1/f8300237c944/request/waveform
Request history	owner/gizmo_g1/f8300237c944/request/history
Write configuration items	owner/gizmo_g1/f8300237c944/write/config
Command to Gizmo	owner/gizmo_g1/f8300237c944/command/direct
Provision	owner/gizmo_g1/f8300237c944/write/provision

The MQTT Client Topics

An MQTT client should Subscribe to the following topic to receive all communications from a specific Gizmo.

owner/gizmo_group/f8300237c944/#

The "#" symbol is a wildcard and replaces the balance of the topic. If used, it must always be the last character in a topic string.

6 Gizmo Communication Specifications (continued from previous page)

6.2 Local Host Data Transport

Gizmo WiFi Sensors also support Local Host data transport, an alternative to MQTT. Local Host transports the same JSON data payload directly via the underlying TCP/IP protocol to Host software. The TCPIP network connection is referred to as data tunnel by which Gizmo and the Host exchange messages.

This Local Host implementation is aimed at simpler, more streamlined systems, operating within a local network, with a single instance of an MMSA Host. Local Host transport eliminates the MQTT protocol layer, MQTT topic overhead, and the required MQTT broker, simplifying messaging and required support services.

Local Host Security

Local Host transport has two levels of security:

1. In its simplest setup, Local Host implements network logon security with a network password. This level does not implement data transport security.
2. In its advanced setup, Local Host implements TLS-based mutual authentication using X.509 certificates to protect and encrypt data in transit. This is the same security used in the MQTT transport above.

Local Host modes are selected during Gizmo provisioning. See the **Gizmo and MMSA Provisioning Guide** for details.

6.3 JSON Data Interchange

All messages published to and from Gizmo shall use the following JSON schema:

Measurement Event Report (

```
{ "Action":"Report",                          //MQTT Subscribe Topic …/report/event
  "GizmoID": "5C027209A1E0",
  "SensorFmwrVer": "1.21",
  "Event": {
     "Type": 0,
     "EventTime": 686951605,
     "Range": 1.25,
     "Temp": 23.7,
     "Volts": 3.4,
     "RSSI": -67,
     "PingUsed": 3,
     "Status": 0,
     "SigStrength": 78,
     "LastEventIndex": 0,
   }
}
```

6 Gizmo Communication Specifications (continued from previous page)

6.3 JSON Data Interchange (continued from previous page)

Configuration Read, Response, or Write

Gizmo configuration is divided into sections. Request and Write tasks can be constructed to access either the entire configuration, one or more sections of configuration, or individual configuration parameters. The following are examples of JSON payloads.

Read Configuration

This JSON payload returns the entire configuration, as shown below.

```
{“Action”:”Request”,                    //MQTT Publish Topic …/request/config
  “Config”:{}
}
```

This JSON payload returns the OpsControl and PingID2 objects

```
{ “Action”:”Request”,                   //MQTT Publish Topic …/request/config
    “Config”:{
        "OpsControl":{},
        "PingID2":{}
      }}
```

Write Configuration

This JSON payload writes just the SleepInterval and OpMode parameters.

```
 { “Action”:”Write”,                    //MQTT Publish Topic …/write/config
   “Config”:{
    "SleepInterval": 3600,
    "OpMode": 1
     }
    }
```

This JSON payload writes Ping 3 control parameters.

```
{ “Action”:”Write”,                     //MQTT Publish Topic …/write/config
  "PingID3": {
    "TxCycles_3": 8,
    "TxHiPwrEnable_3": 1,
    "BlankRange_3": 2.5,
    "GainSwitch_3": 12.6,
    "EndOfRange_3": 40.0,
    "Threshold_3A": [85, 75, 50, 40],
    "ThreshDistance_3A": [5.50, 10.7, 15.0]}
}
```

6 Gizmo Communication Specifications (continued from previous page)

6.3 JSON Data Interchange (continued from previous page)

Configuration Payload

The following is the entire Gizmo configuration payload or the Config object.
See the Configuration Details table for value types and limits.


```
{
 "GizmoID": “5C027209A1E0”,
 "Config": {
  "SensorInfo": {
   "Model": "M5510-150kHz",
   "SN": 9077243,
   " SensorFmwrVer ": 1.02,
   "SensorHardware": 1.00,
   "RadioFmwrVer": 2.05,
   "SensorTime": 686951605,
   "EventCount": 16,
   "StartIndex": 0,
   "HostType": 0,
   "HostDirecetSecurity": 0,
   "HostPayload": 0,
   "RadioType": 1,
   "ErrorIndication": 0
  },
  "OpsControl": {
   "SleepInterval": 3600,
   "OpMode": 1,
   "TimeZone": 1,
   "AlarmLow": 15.7,
   "AlarmHigh": 81.2,
   "Description": "32 char string",
   "RadioConnectTO": 45,
   "InterMessageTO": 3,
   "ProvisioningTO": 300,
   "HostType": 1,
   "HostDirectSecurity": 0,
   "HostProtocol": 0,
   "ReportCount": 4
  },
  "MeasureControl": {
   "PresetTempOffOn": 0,
   "PresetTempValue": 22.5
  },
  "PingID1": {
   "TxCycles_1": 8,
    "TxHiPwrEnable_1": 0,
    "BlankRange_1": 2.5,
    "GainSwitch_1": 24.6,
    "EndOfRange_1": 24.0,
    "Threshold_1A": [85, 75, 50, 40],
    "ThreshDistance_1A": [5.50, 10.7, 15.0]
  },
```

6 Gizmo Communication Specifications (continued from previous page)

6.3 JSON Data Interchange (continued from previous page)

Configuration Payload (continued from previous page)


```
  "PingID2": {
    "TxCycles_2": 8,
    "TxHiPwrEnable_2": 0,
    "BlankRange_2″: 2.5,
    "GainSwitch_2": 24.6,
    "EndOfRange_2": 24.0,
    "Threshold_2A": [85, 75, 50, 40],
    "ThreshDistance_2A": [5.50, 10.7, 15.0]
  },
  "PingID3": {
    "TxCycles_3": 8,
    "TxHiPwrEnable_3": 1,
    "BlankRange_3": 2.5,
    "GainSwitch_3": 24.6,
    "EndOfRange_3": 24.0,
    "Threshold_3A": [85, 75, 50, 40],
    "ThreshDistance_3A": [5.50, 10.7, 15.0]
  },
  "PingID4": {
    "TxCycles_4": 8,
    "TxHiPwrEnable_4": 1,
    "BlankRange_4": 2.5,
    "GainSwitch_4": 54.6,
    "EndOfRange_4": 28.7,
    "Threshold_4A": [85, 75, 50, 40],
    "ThreshDistance_4A": [5.50, 10.7, 15.0]
  },
 }
}
```

6 Gizmo Communication Specifications (continued from previous page)

6.3 JSON Data Interchange (continued from previous page)

History Request/Response

History Request

Request history with the number of event records and the index of the first event requested. Gizmo will return the event records requested.

```
{  “Action”:”Request”,                    //MQTT Publish Topic …/request/history
  “History”:{
    "RequestCount": 20,
    "StartIndex": 51
  }
}
```

History Response (MQTT Subscribe Topic …/response/history)

History response includes one or more data packets with up to 14 event array objects in each packet. The HistoryDetails returns the report StartIndex, RequestCount, and the data Packet Count for this report.

```
{ “Action”:”Response”,                    //MQTT Subscribe Topic …/response/history
    "GizmoID": “5C027209A1E0”,
    "HistoryDetails": {
      “StartIndes”: 51,
      "RequestCount": 20,
      "PacketCount": 2
    }
}
```

History event records response

```
{  “Action”:”Response”,                   //MQTT Subscribe Topic …/response/history
  "GizmoID": “5C027209A1E0”,
  "HistoryData": [
    {
      "EventTime": 0,
      "Range": 68,
      "Temp": 24.3,
      "Volts": 3.5,
      "RSSI": -72,
      "Status": 8,
      "SigStrength": 78
    },
    {
      "EventTime": 0,
      "Range": 68,
      "Temp": 24.3,
      "Volts": 3.5,
      "RSSI": -72,
      "Status": 8,
      "SigStrength": 78
    }
  ]}
```

6 Gizmo Communication Specifications (continued from previous page)

Commands

Gizmo subscribes to the topic "…/command/direct". The client should publish one of the following JSON objects to the MQTT Publish Topic "…/command/direct" to execute the command.

```
{ "Action":"Command",                    //MQTT Publish Topic "…/command/direct"
  "Command":{
    "Reset": {}
  }
}
```

Command Function	JSON Object
Gizmo Reset	"Reset":{}
Clear Status	"ClearStatus":{}
Clear History	"ClearHistory":{}
Restore Factory Defaults	"RestoreFactoryDefaults":{}
Update Sensor Firmware	"SensorFOTA":{}
Update Radio Firmware	"RadioFOTA":{}

Waveform Request

Waveform includes ping ID, temperature, status, and up to 2000 waveform data points. Waveform collection begins with a ping selected from one of the four ping IDs programmed in config below

This triggers a ping using Ping ID 2

```
{  "Action":"Request",                   //MQTT Publish Topic …/request/waveform
    "WaveformPing": {
      "WFPing": 2
    }
  }
```

Note: *Pings are numbered 1 thru 4. Use 5 to retrieve all 4 ping waveforms with one command.*

Waveform Response

Waveform includes up to 2000 waveform data points depending on waveform ping configuration. The waveform request returns the WaveformDetails object followed by WaveformData objects. WaveformDetails includes following.

```
{  "Action":"Response",          //MQTT Subscribe Topic …/response/waveform
     "GizmoID":   "5C027209A1E0",
    "WaveformDetails": {
     "WFPingUsed": 2,
     "WFTemp": 23.7,
     "WFStatus": 0,
     "WFSampleRate": 25.3,
"WFSampleCount": 1440,
     "WFBase": 47,
     "WFPacketCount": 1356,
     "BatV": 3.53
   }}
```

6 Gizmo Communication Specifications (continued from previous page)

The waveform response includes one or more WaveformData objects which includes the WFSampleIndex and up to 400 waveform samples. The WFSampleIndex is the block position in the total waveform data array.

```
{ “Action”:”Response”,          //MQTT Subscribe Topic …/response/waveform
 "GizmoID": “5C027209A1E0”,
 “WaveformData”: {
    "WFSampleIndex": 0,
    "WFData": [ 0, 1, 2, 4, 5, 6, 7, ....., 399 ]
 }}

{ “Action”:”Response”,          //MQTT Subscribe Topic …/response/waveform
 "GizmoID": “5C027209A1E0”,
  “WaveformData”: {
    "WFSampleIndex": 400,
    "WFData": [ 0, 1, 2, 4, 5, 6, 7, ....., 399 ]
 }}
```

Note: 1. If WiFi network is dynamic, only SSID, SecurityType, and SecurityKey are required.
2. Use of Host UserName and Password depends on broker requirements.

7 Gizmo Provisioning Overview

Provisioning data is the set of connection and security information that enables Gizmo to connect to a network, connect to a remote host or broker, and transfer data securely over that connection. Gizmo provides two provisioning profiles, Primary and Fallback. Both provisioning profiles are stored in Gizmo's internal non-volatile memory.

Provisioning Details

Gizmo uses a two-layer provisioning strategy:

- The Network Connect Layer provides details of network connection and login, as well as host connection. At a minimum Gizmo requires Network Connect provisioning. In the case of Gizmo WiFi, Network Connect details include the WiFi network SSID and Password as well as the host (broker) IP / URL and Port.
- The Security Layer provides the x.509 Device Certificate, Private Key, and the root Certificate Authority. Gizmo uses the credentials in this layer to mutually authenticate connection and transfer encrypted data to sites like AWS IoT Core.

Provisioning Methods

Gizmo, support software, and services offer 3 ways to provision Gizmo.

- Direct Provisioning.
  In this method Gizmo Provisioning software connects and transfers provisioning details using Gizmo's built in WiFi Access Point (AP).
  - The user first enters provisioning details via Gizmo Provisioning software GUI
  - The user manually opens Gizmo's WiFi provisioning AP
  - Gizmo Provisioning software connects to the AP and sends provision details.
  - Gizmo stores provisioning details in its non-volatile memory for use from then on.

- Fleet Provisioning
  In this method provisioning is accomplished via an Automated Fleet Provisioning service
  - Gizmo is factory pre-provisioned with default or bootstrap Claim security credentials and network details.
  - The user will enter final provisioning details for Gizmo's installation site via the Gizmo Provisioning web site
  - When Gizmo is powered on site, it connects to the fleet provisioning service using bootstrap credentials provided at the factory.
  - This connection triggers the download of final site provisioning details.
  - Gizmo stores provisioning details in its non-volatile memory for use from then on.

- Provisioning via MQTT
  Once Gizmo is initially provisioned and communicates via MQTT, provisioning details can be updated by sending all or part of the JSON Provisioning object.

  See the **Gizmo and MMSA Provisioning Guide** for details.

8 Gizmo Data Registers

Measurement Event Report

Data Item	Data Type	Description
Type	Int	Event reason 0 = none 1 = Report on every wakeup 2 = Report after delayed count events 3 = Report catchup 4 = Report on alarm 5 = Report on fill or empty target reached 6 = Report on override timeout 7 = Report on override sample ready 8 =Test report after provisioning 9 = Report on request
Time	uint32	Event time stamp, seconds in J2000 epoch time
Range	Float	Measurement range in inches
Temp	Float	Temperature in degrees C
Volts	Float	Gizmo battery voltage
RSSI	Int	Radio signal
PingUsed	Int	Value from 1 to 4 indication which ping successfully returned the range measurement
Status	Int	Bit 0: Sensor Detection Fault. Bit 1: Temperature Probe Fault. Bit 2: Battery Low. Replace. Bit 4-6: Communication type 0 = MQTT 1 = MQTT with Fleet Provision 2 = Local Host 3 = Local Host with TLS Bit 7: Gizmo needs RTC time set Bit 13: Config validate error Bit 14: Config set to Factory Defaults Bit 15: Config set to Embedded defaults Bit 16: Radio Provision File Error Bit 17: Sensor FOTA Error Bit 18: Radio FOTA Error
SignalStrength	Int	Ultrasonic signal strength in percent of Full scale
LastEventIndex	Int	Start index of current block of history data

8 Gizmo Data Registers (continued from previous page)

Waveform Details

Header object applies to waveform data to follow.

Data Item	Data Type	Limits	Description
Ping Used	Int	1 to 4	Ping number used to collect waveform data
Temperature	float	Deg C	Temperature when waveform was collected
Status	Int		NA
SampleRate	float	0.0 to 100.0uS in 0.1uS	Waveform ADC sample rate
SampleCount	Int	2000 max	Ping's End of Range time / SampleRate
WFBase	int	0 to 100	Value of measurement baseline voltage in percent of full scale
WFPacketCount	int		Total number of waveform data packets returned in this request

Waveform Data

One or more blocks of data.

Data Item	Data Type	Limits	Description
SampleIndex	Int		Waveform data array offset for this block
Data	Int[400]	0 to 255	Up to 400 sample array

Configuration Details

Data Item	Data Type	Limits	Default	Description
Sensor Info				**Sensor Info Section**
Model	String	Read Only	-	Gizmo model number
SN	Uint32	Read Only	-	Gizmo serial number
SensorFmwrVer	Float	Read Only	-	Sensor firmware revision
RadioFmwrVer	Float	Read Only	-	Radio firmware revision
SensorPCBVer	Float	Read Only	-	Sensor PCB revision
SensorTime	Uint32	J2000 epoch time	2451545.0	Gizmo time, seconds in J2000 epoch GMT time
EventCount	Uint16	Read Only	-	Available history events
RadioType	Uint8	Read Only	1	Bits 1-3: WiFi = 1 Cellular = 2 Bits 4-7: 0 = MQTT 1 = MQTT with Fleet Provision 2 = Local Host 3 = Local Host with TLS

8 Gizmo Data Registers (continued from previous page)

Configuration Details (continued from previous page)

Data Item	Data Type	Limits	Default	Description
Sensor Info				**Sensor Info Section**
Sensor Status	Uint16		-	Bit 0: Sensor Detection Fault. Bit 1: Temperature Probe Fault. Bit 2: Battery Low. Replace. Bit 4-6: Communication type 0 = MQTT 1 = MQTT with Fleet Provision 2 = TCP Direct 3 = TCP Direct with TLS Bit 7: Gizmo needs RTC time set Bit 13: Config validate error Bit 14: Config set to Factory defaults Bit 15: Config set to Embedded defaults Bit 16: Radio Provision File Error Bit 17: Sensor FOTA Error Bit 18: Radio FOTA Error
OpsControl				Operation Control Section
SleepInterval	Uint32	60 sec. to 7 days	60	Sleep interval
OpMode	Uint8	0 to 5	1	Mode 1=Report on every wakeup 2=Report after history count measurements. 3=Report all events since last report
TimeZone	Uint8		0	Gizmo time zone (not used)
AlarmLow	Float	0 - 120 in (150) 0 - 250 in (95)	98″ (150) 216″ (95)	Distance below which, measure is in alarm
AlarmHigh	Float	0 - 120 in (150) 0 - 250 in (95)	11″ (150) 24″ (95)	Distance above which, measure is in alarm
Description	String	32 – 126 (ASCII character limit)	All 32 (space)	32 user description
RadioConnectTO	Uint16	5 to 300 seconds	45	Max radio attempt to connect wait time before sleep

8 Gizmo Data Registers (continued from previous page)

Configuration Details (continued from previous page)

Data Item	Data Type	Limits	Default	Description
OpsControl				Operation Control Section
InterMessageTO	Uint16	2 to 60 sec	5	Max inter-message time before sleep
ProvisioningTO	Uint16	60 to 600 sec	600	Max provisioning time
ReportCount	Uint8	1 to 255	1	Number of wake-up, measurement, and sleep cycles before report
Measurement Control				
Preset Temperature Off / On	Boolean	0 or 1	0	0=Off, 1=On
Preset Temperature Value	Float	-30 to +70	23	Degree C
TempCal	Float	-50 to +50	Read only	Temperature calibration offset by factory
Ping1				Config: Ping 1 Section
TxCycles	Uint8	1 - 4 cycles	1 (150) 2 (95)	Number cycles in transmit tone burst
TxHiPwrEnable	Boolean	0 or 1	0	0= low power, 1= high power
BlankRange	Float	0 - 120 in (150) 0 - 250 in (95)	3.8 (150) 7.8 (95)	Measurement blanking range in inches
GainSwitch	Float	0 - 120 in (150) 0 - 250 in (95)	10.0 (150) 16.0 (95)	Distance for low to high gain switch in inches
EndOfRange	Float	0 - 120 in (150) 0 - 250 in (95)	120 (150) 250 (95)	End of range in inches
Threshold	Uint8[4]	0 - 100 %	60,45, 35,30	Array of 4 thresholds, percentage of maximum echo amplitude values
ThreshDistance	Float[3]	0 - 120 in (150) 0 - 250 in (95)	12,15,18 (150) 16,20,24 (95)	Array of 3 threshold switch distances in milliseconds. First element is time of switch between threshold 1 and threshold 2

8 Gizmo Data Registers (continued from previous page)

Configuration Details (continued from previous page)

Data Item	Data Type	Limits	Default	Description
Ping2				Config: Ping 2 Section
TxCycles	Uint8	0 - 32 cycles	10 (150) 10 (95)	Number cycles in transmit tone burst (to disable ping, set to 0)
TxHiPwrEnable	Boolean	0 or 1	0	0= low power, 1= high power
BlankRange	Float	0 - 120 in (150) 0 - 250 in (95)	6.0 (150) 12.0 (95)	Measurement blanking range in inches
GainSwitch	Float	0 - 120 in (150) 0 - 250 in (95)	16.0 (150) 24.0 (95)	Distance for low to high gain switch in inches
EndOfRange	Float	0 - 120 in (150) 0 - 250 in (95)	120 (150) 250 (95)	End of range in inches
Threshold	Uint8[4]	0 - 100 %	60,45, 35,30	Array of 4 thresholds, percentage of maximum echo amplitude values
ThreshDistance	Float[3]	0 - 120 in (150) 0 - 250 in (95)	12,15,18 (150) 16,20,24 (95)	Array of 3 threshold switch distances in milliseconds. First element is time of switch between threshold 1 and threshold 2
Ping3				Config: Ping 3 Section
TxCycles	Uint8	0 - 32 cycles	10 (150) 10 (95)	Number cycles in transmit tone burst (to disable ping, set to 0)
TxHiPwrEnable	Boolean	0 or 1	1	0= low power, 1= high power
BlankRange	Float	0 - 120 in (150) 0 - 250 in (95)	8.0 (150) 15.0 (95)	Measurement blanking range in inches
GainSwitch	Float	0 - 120 in (150) 0 - 250 in (95)	16.0 (150) 24.0 (95)	Distance for low to high gain switch in inches
EndOfRange	Float	0 - 120 in (150) 0 - 250 in (95)	120 (150) 250 (95)	End of range in inches
Threshold	Uint8[4]	0 - 100 %	60,45, 35,30	Array of 4 thresholds, percentage of maximum echo amplitude values
ThreshDistance	Float[3]	0 - 120 in (150) 0 - 250 in (95)	15,18,21 (150) 18,22,26 (95)	Array of 3 threshold switch distances in milliseconds. First element is time of switch between threshold 1 and threshold 2

8 Gizmo Data Registers (continued from previous page)

Configuration Details (continued from previous page)

Data Item	Data Type	Limits	Default	Description
Ping4				Config: Ping 4 Section
TxCycles	Uint8	0 - 32 cycles	20 (150) 20 (95)	Number cycles in transmit tone burst (to disable ping, set to 0)
TxHiPwrEnable	Boolean	0 or 1	1	0= low power, 1= high power
BlankRange	Float	0 - 120 in (150) 0 - 250 in (95)	8.0 (150) 15.0 (95)	Measurement blanking range in inches
GainSwitch	Float	0 - 120 in (150) 0 - 250 in (95)	16.0 (150) 24.0 (95)	Distance for low to high gain switch in inches
EndOfRange	Float	0 - 120 in (150) 0 - 250 in (95)	120 (150) 250 (95)	End of range in inches
Threshold	Uint8[4]	0 - 100 %	60,45, 35,30	Array of 4 thresholds, percentage of maximum echo amplitude values
ThreshDistance	Float[3]	0 - 120 in (150) 0 - 250 in (95)	15,18,21 (150) 18,22,26 (95)	Array of 3 threshold switch distances in inches. First element is time of switch between threshold 1 and threshold 2
Fill Empty Override				Config: Fill Empty Section
StartTime	Uint32	J2000 epoch time	2459946.0000000	Override start time, seconds in J2000 epoch time (1-1-2023)
OverrideMode	Uint8	0 - 3	0	0= No Override, 1= Fill, 2= Empty, 3=Fast Sample
OverrideTarget	Float	0 - 120 in (150) 0 - 250 in (95)	0	Target, when reached Gizmo will send a report and terminate mode.
OverrideRate	Uint16	0 - 600 seconds	0	Override mode sample rate (override sleep time)
OverrideTimeout	Uint16	0 - 36000 seconds	600	Maximum override mode time. Reverts to normal sleep time after this time expires.

Revision History

Revision	Date	Notes
1.0	June 6, 2024	First release