# User Guide

## Sentrius™ BT510

*Version 3.0*

# Revision History

| Version | Date | Notes | Contributor | Approver |
|---|---|---|---|---|
| 1.0 | 03 Feb 2020 | Initial Release | Brent Mikkelsen<br>Ian Tracy | Jonathan Kaye |
| 1.1 | 04 Feb 2020 | Added Japan and AS/NZS Regulatory statements.<br>Added ordering information | Dave Neperud | Jonathan Kaye |
| 1.2 | 03 Mar 2020 | Updated Record Event Types table information for clarity | Brent Mikkelsen | Jonathan Kaye |
| 1.3 | 20 Mar 2020 | Updated mobile App section. Added more information to the Advertisement con | Brent Mikkelsen | Jonathan Kaye |
| 1.4 | 16 April 2020 | Provided more detail on active mode. Changed default values for advertising duration. Added more reset information. | Brent Mikkelsen | Jonathan Kaye |
| 1.5 | 21 May 2020 | Added a line 13 (RESERVED) to Table 5 | Brent Mikkelsen | Jonathan Kaye |
| 2.0 | 11 Nov 2020 | Reset is not required with version 4.0.0 or later.<br>Update read log example with C-structure. Add link to Python examples.<br>Updated all regulatory information | Andrew Hedin<br>Maggie Teng<br>Ryan Urness | Jonathan Kaye |
| 2.1 | 10 Mar 2021 | Update for version 5.1 of firmware | Andrew Hedin | Chris Boorman |
| 2.2 | 11 Mar 2021 | Transferred all detailed regulatory information to a separate document | Sue White | Jonathan Kaye |
| 2.3 | 22 Apr 2021 | Updated advertisements table notes. Add to FAQ. | Andrew Hedin | Chris Boorman |
| 2.4 | 9 June 2021 | Updated for version 5.3. Add recommendations for operation in cold environments. | Andrew Hedin | Jonathan Kaye |
| 3.0 | 5 May 2025 | Ezurio Rebranding | Sue White | Dave Drogowski |

# Contents

# 1 About this Guide

This document provides a comprehensive guide on how to configure the Sentrius™ BT510 Sensor functionality, including Bluetooth settings and temperature readings, motion, and contact detection.

# 2 Introduction

## 2.1 Product Overview

The Sentrius™ BT510 Sensor is a battery powered, Bluetooth v5 long-range integrated sensor platform enabling robust, reliable sensor data transfer in the harshest of environments. It contains a temperature sensor, also seamlessly integrating open/close contact and motion/impact detection and BLE beaconing capabilities. The BT510 is powered by Ezurio's field proven BL654 BLE module that integrates Nordic Semiconductor's nRF52840 SoC silicon.


1. Fixing holes
2. LED
3. Hidden button to wake device
4. Magnetic switch

*Figure 1: Top of the Sentrius™ BT510 sensor*

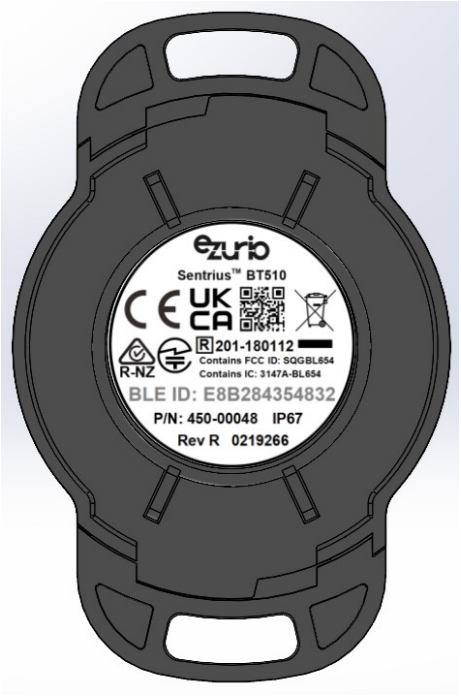



*Figure 2: Back of the Sentrius™ BT510 sensor*

**Note:** Ezurio has a comprehensive staff of design services engineers available to help customize the sensor. Please contact your local Ezurio sales representative for more details.

## 2.2 Specifications

See the BT510 product brief for detailed specifications. It's available from the documentation tab of the BT510 series product page: https://www.ezurio.com/bt510

# 3 Device Operation

## 3.1 Activating the Sensor

The **Sentrius™** BT510 does not have a power switch. It is shipped with an installed battery and is in a low power state called shelf mode.

To wake the device, firmly press the button in the center of the round face for at least three seconds until the green LED turns on. This puts the device in active mode. Once in active mode, the BT510 is able to report back the sensor readings for the magnet, accelerometer, and temperature.

When you release the button, the green LED blinks once a second.

The sensor is in sleep mode from the factory. Pressing the button for three seconds wakes the sensor and it starts to advertise. The sensor is now ready for normal operation configuration. This can be done using a phone or gateway. The LED blinks for 30 seconds or until a connection is made, the button is pressed again, or the sensor stops advertising. The default state of the sensor is to advertise for 30 minutes.

## 3.2 Battery Check

Quickly press the button to perform a battery good check. If the battery is good, the sensor briefly blinks the green LED and starts to advertise for 15 seconds. If the battery is below the recommended operating voltage, then the LED does not blink but the sensor tries to advertise. It is possible for the mobile application to connect to the BT510 within the 15 second advertisement duration, but if the BT510 is not set to active mode the sensor does not take any sensor readings.

## 3.3 Factory Reset

If the button is held for more than 10 seconds and released, the sensor performs a factory reset. A factory reset erases all logs, pairing information, and resets the configuration to its default state. It takes the BT510 two seconds after the reset before it is running again, and a button press can be detected.

When the button (as shown in Figure 1) is held for 10 seconds, the LED turns yellow or red. If the LED turns red, then factory reset is not allowed using the button. This is because the sensor is locked. The sensor can be locked by using one of the JSON properties. When locked, the factory reset can still be performed using Bluetooth. If the LED turns yellowish orange, then factory reset is allowed. The yellowish orange LED blinks when the button is released. Once the BT510 is reset to factory settings, the BT510 sensor must be put back into active mode by holding the button for three seconds.

## 3.4 Replacing Batteries

The battery is a 3-volt lithium of CR2477X type.

---

**Note:** The battery door cover has a gasket inside to keep out liquids.

---

## 3.5 Care and Maintenance

The sensor can be cleaned with a mild, non-abrasive detergent. Because it is not waterproof, do not immerse it in water.

The sensor does not require any calibration.

# 4 Sensor Architecture

The **Sentrius™** BT510 advertises events. An event can be a temperature measurement, an alarm, a battery measurement, a button press, a door opening/closing, or movement. The configuration of a sensor determines what kind of events it generates.

You can configure a sensor using a Bluetooth connection and the Ezurio virtual serial port (vSP) service. The protocol sent over the virtual serial port is JSON-RPC version 2. This allows you to add new commands and features without changing the Bluetooth interface. More details can be found at https://www.jsonrpc.org/specification.

## 4.1 Advertisements

Advertising each event once for a set duration is the default behavior of the sensor. This allows time for a gateway to detect an event. Once a sensor is configured, it is possible that a gateway never needs to connect to a sensor. If multiple events occur at the same time, the sensor queues events and advertises each one for the configured advertising duration.

If the advertising duration is set to zero, the sensor will advertise each event for 15 x the advertising interval and will advertise the last event indefinitely.

If the advertising duration is non-zero, the sensor will advertise each event for the advertising duration and will stop advertising when there aren't any new events.

For each new event, the record number increments. You can use this value to filter out duplicate advertisements.

The TLV (type-length-value) fields are shown in the order that is broadcast by the sensor. However, this order is not guaranteed.

The sensor can be configured to transmit with an output power of +8 dBm. It can also be configured to use the LE coded PHY.

When using the coded PHY, the scan response isn't used.

### 4.1.1 1M PHY

*Table 1: 1M PHY*

| Byte | Description | Value/Notes |
|---|---|---|
| 0 | 0x02 | Length (0x02) |
| 1 | GAP_ADTYPE_FLAGS | Type (0x01) |
| 2 | GAP_ADTYPE_FLAGS_BREDR_NOT_SUPPORTED | Data |
| 3 | 0x1b (27) | Length (length is not included in overall length) |
| 4 | GAP_ADTYPE_MANUFACTURER_SPECIFIC | 0xFF (Type) |
| 5 | Company ID 1 | 0x77 (Ezurio) |
| 6 | Company ID 2 | 0x00 |
| 7 | Protocol ID LSB | 0x01 (Identifies the advertisement format) |
| 8 | Protocol ID MSB | 0x00 |
| 9 | Network ID LSB | Assigned during configuration. Default is 0x000. |
| 10 | Network ID MSB | This can be used for filtering advertisements |
| 11 | Flags LSB | Indicate the current state of the system |
| 12 | Flags MSB | See 4.1.6 Flags |
| 13 | BD_ADDR 0 | Random static Bluetooth address (0 is LSB) |
| 14 | BD_ADDR 1 | |
| 15 | BD_ADDR 2 | |
| 16 | BD_ADDR 3 | |
| 17 | BD_ADDR 4 | |
| 18 | BD_ADDR 5 | MSB |
| 19 | Record Type | See 4.1.4 Record Event Types |
| 20 | Record Number LSB | This count matches the index in the NV log. It will rollover |
| 21 | Record Number MSB | |
| 22 | Epoch 0 LSB | Seconds since client's reference point; used to timestamp data; tested with reference of January 1, 1970 midnight UTC. |
| 23 | Epoch 1 | |
| 24 | Epoch 2 | |
| 25 | Epoch 3 MSB | |
| 26 | Data byte 0 LSB | See Table 4 to match record type to data |

| Byte | Description | Value/Notes |
|---|---|---|
| 27 | Data byte 1 | |
| 28 | Data byte 2 | |
| 29 | Data byte 3 (MSB) | |
| 30 | Reset Count LSB | For testing purposes. |

### 4.1.2 1M PHY Scan Response

*Table 2: 1M PHY scan response*

| Byte | Description | Value/Notes |
|---|---|---|
| 0 | 0x10 (16) | Length (length is not included in overall length) |
| 1 | GAP_ADTYPE_MANUFACTURER_SPECIFIC | 0xFF (Type) |
| 2 | Company ID 1 | 0xE4 (Ezurio) |
| 3 | Company ID 2 | 0x00 |
| 4 | Protocol ID LSB | 0x03 (Identifies the advertisement format) |
| 5 | Protocol ID MSB | 0x00 |
| 6 | Product ID LSB | Identification of the device transmitting |
| 7 | Product ID MSB | |
| 8 | Firmware Version Major | Version of the main application installed |
| 9 | Firmware Version Minor | |
| 10 | Firmware Version Patch | |
| 11 | Firmware Type | Firmware type can be used to interleave versions for multiple devices |
| 12 | Configuration Version | Configuration version is updated when advertisement is built |
| 13 | Boot Loader Version Major | |
| 14 | Boot Loader Version Minor | |
| 15 | Boot Loader Version Patch | |
| 16 | Hardware Version | |
| 17 | Length (X) | (< =13) |
| 18 | DEVICE_NAME | 0x08 or 0x09 |
| 19 | | Max Complete Name (0x09) is 12 characters |
| 20 | | |
| 21 | | |
| 22 | | |
| 23 | | |
| 24 | | |
| 25 | | |
| 26 | | |
| 27 | | |
| 28 | | |
| 29 | | |
| 30 | | |

### 4.1.3 LE Coded PHY

*Table 3: LE Coded PHY*

| Byte | Description | Value/Notes |
|---|---|---|
| 0 | 0x02 | Length (0x02) |
| 1 | GAP_ADTYPE_FLAGS | Type (0x01) |
| 2 | GAP_ADTYPE_FLAGS_BREDR_NOT_SUPPORTED | Data |
| 3 | 0x26 (38) | Length (length is not included in overall length) |
| 4 | GAP_ADTYPE_MANUFACTURER_SPECIFIC | 0xFF (Type) |
| 5 | Company ID 1 | 0x77 (Ezurio) |
| 6 | Company ID 2 | 0x00 |
| 7 | Protocol ID LSB | 0x02 (Identifies the advertisement format) |
| 8 | Protocol ID MSB | 0x00 |
| 9 | Network ID LSB | Assigned during configuration. Default is 0x000. |
| 10 | Network ID MSB | This can be used for filtering advertisements |
| 11 | Flags LSB | Indicate the current state of the system |
| 12 | Flags MSB | See 4.1.6 Flags |
| 13 | BD_ADDR 0 | Random static Bluetooth address (0 is LSB) |
| 14 | BD_ADDR 1 | |
| 15 | BD_ADDR 2 | |
| 16 | BD_ADDR 3 | |
| 17 | BD_ADDR 4 | |
| 18 | BD_ADDR 5 | MSB |
| 19 | Record Type | See 4.1.4 Record Event Types |
| 20 | Record Number LSB | This count matches the index in the NV log. It will rollover |
| 21 | Record Number MSB | |
| 22 | Epoch 0 LSB | Seconds since client's reference point; used to timestamp data; tested with reference of January 1, 1970 midnight UTC. |
| 23 | Epoch 1 | |
| 24 | Epoch 2 | |
| 25 | Epoch 3 MSB | |
| 26 | Data 0 LSB | See Table 4 to match record type to data |
| 27 | Data 1 | |
| 28 | Data 2 | |
| 29 | Data 3 (MSB) | |
| 30 | Reset Count LSB | For testing purposes. |
| 31 | Product ID LSB | Identification of the device transmitting |
| 32 | Product ID MSB | |
| 33 | Firmware Version Major | Version of the main application installed |
| 34 | Firmware Version Minor | |
| 35 | Firmware Version Patch | |
| 36 | Firmware Type | Firmware type can be used to interleave versions for multiple devices |
| 37 | Configuration Version | Configuration version is updated when advertisement is built |

| Byte | Description | Value/Notes |
|---|---|---|
| 38 | Boot Loader Version Major | |
| 39 | Boot Loader Version Minor | |
| 40 | Boot Loader Version Patch | |
| 41 | Hardware Version | |
| 42 | Length (X) | (<=24) |
| 43 | DEVICE_NAME | 0x09 |
| 44 | | |
| ... | | |
| 66 | | |

### 4.1.4 Record Event Types

*Table 4: Record event types*

| ID | Event | Data | Format |
|---|---|---|---|
| 0 | RESERVED | - | |
| 1 | TEMPERATURE | TEMPERATURE | Hundredths of degree C (signed 16-bit number) |
| 2 | MAGNET (PROXIMITY) | MAGNET STATE | See 4.1.5 Magnet States |
| 3 | MOVEMENT | - | - |
| 4 | ALARM HIGH TEMP 1 | TEMPERATURE | Hundredths of degree C |
| 5 | ALARM HIGH TEMP 2 | TEMPERATURE | Hundredths of degree C |
| 6 | ALARM HIGH TEMP CLEAR | TEMPERATURE | Hundredths of degree C |
| 7 | ALARM LOW TEMP 1 | TEMPERATURE | Hundredths of degree C |
| 8 | ALARM LOW TEMP 2 | TEMPERATURE | Hundredths of degree C |
| 9 | ALARM LOW TEMP CLEAR | TEMPERATURE | Hundredths of degree C |
| 10 | ALARM DELTA TEMP | TEMPERATURE | Hundredths of degree C |
| 12 | BATTERY GOOD | BATTERY VOLTAGE | Millivolts (unsigned 16-bit number) |
| 13 | ADVERTISE ON BUTTON | BATTERY VOLTAGE | Millivolts |
| 14 | RESERVED | - | - |
| 15 | RESERVED | - | - |
| 16 | BATTERY BAD | BATTERY VOLTAGE | Millivolts |
| 17 | RESET | RESET REASON | See 4.1.7 Reset Reason |

### 4.1.5 Magnet States

The magnet states are NEAR = 0 (door closed) and FAR = 1 (door open).

### 4.1.6 Flags

The flags are a bitmask of the current state of the sensor.

***Table 5: Flags***

| Bit | Name |
|---|---|
| 0 | RTC was set (epoch) |
| 1 | Active Mode |
| 2 | Any flag was set |
| 3 | RESERVED |
| 4 | RESERVED |
| 5 | RESERVED |
| 6 | RESERVED |
| 7 | Low Battery Alarm |
| 8 | high temperature alarm bit 0 |
| 9 | high temperature alarm bit 1 |
| 10 | low temperature alarm bit 0 |
| 11 | low temperature alarm bit 1 |
| 12 | Delta Temperature Alarm |
| 13 | RESERVED |
| 14 | Movement Alarm |
| 15 | Magnet State |

### 4.1.7 Reset Reason

The reset reasons come from the RESETREAS register in the Nordic nRF52840. Reserved values do not apply to the sensor.

***Table 6: Reset reasons***

| Value | Reason |
|---|---|
| 0 | Power Up |
| 1 | Reset Pin |
| 2 | Watchdog |
| 3 | Software Request |
| 4 | CPU Lock-up detected |
| 5 | Reserved |
| 6 | Low Power Comparator |
| 7 | Reserved |
| 8 | Reserved |
| 9 | Reserved |
| 10 | Unknown |

## 4.2 Ezurio vSP (Virtual Serial Port)

Details can be found at https://www.ezurio.com/documentation/application-note-using-vsp-smartbasicpdf.

The sensor does not use the optional Modem In/Out characteristics.

## 4.3 JSON-RPC

The sensor supports version 2.0 of the specification. The sensor does not support batch commands.

More details can be found at https://www.jsonrpc.org/specification.

### 4.3.1 Methods

*Table 7: JSON-RPC methods*

| Method (Command) | Parameters | Description |
|---|---|---|
| get | ‹properties› | The get command is used to get the value of an attribute. |
| set | ‹properties› | The set command is used to set attributes. |
| dump | None for all<br>0 for read-only<br>1 for read-write | Used to read all attributes. Configurable attributes are read-write. |
| reboot | None or 0 for normal<br>Non-zero for enter bootloader | Restart the sensor immediately after sending acknowledgement. |
| factoryReset | NA | Reset the device to factory settings |
| prepareLog | integer, positional | 0 = FIFO mode, 1 = LIFO mode |
| readLog | integer, positional | Number of events to read |
| ackLog | integer, positional | Number of events to invalidate |
| setEpoch | integer, positional | Seconds since client's reference point; used to timestamp data; tested with reference of January 1, 1970 midnight UTC. |
| getEpoch | none | Returns epoch in result |
| ledTest | integer, duration of each step | Both LEDs off for X milliseconds<br>Green LED on for X milliseconds<br>Red LED on for X milliseconds<br>Both LED on for X milliseconds.<br>Processor remains running (doesn't enter low power mode) for duration of test. The minimum duration is 10 milliseconds. This command is primarily for production test. |

## 4.4 Sensor Configuration

The sensor has multiple properties (attributes) that can be used to configure it for a particular use case.

The read only parameters reflect the current state of the sensor. Issuing a JSON-RPC command to get the temperature does not cause the sensor to take a temperature measurement. It returns the value of the last temperature measurement taken.

As indicated in the table, some parameters require the sensor to be reset before they take effect. For versions of firmware prior to 4.0.0, the software reset command will need to be sent before these changes to the parameter will work properly.

**Note:** The sensor contains more parameters than those that are listed in the table. They are used during production and reliability testing by Ezurio. They may not exist in future software versions.

### 4.4.1 Example

In this example system, the device name is changed during configuration so that unconfigured devices are the only devices that advertise with the name *BT510*.

1. Start Scan for *BT510*.

```
Found Device E235431D1F8E with RSSI -47 and advertisement data
"0201061BFF7700010000000808E1F1D4335E20C030008B1D55DB80B00000210FFE4000300000001030A000000000008
06094254353130"
```

2. Stop Scan.
3. Connect to E235431D1F8E.
4. Set time in sensor.

```
>> {"jsonrpc": "2.0", "method": "setEpoch", "params": [1574285874], "id": 2}

<< {"jsonrpc": "2.0", "id": 2, "result": "ok"}
```

5. Configure sensor to take a temperature measurement every 15 minutes, a battery measurement every hour, and enable motion detection.

```
>> {"jsonrpc": "2.0", "method": "set", "params": {"sensorName": "Test-02", "location": "desk",
"advertisingDuration": 15000, "batterySenseInterval": 3600, "temperatureSenseInterval": 900,
"odr": 5}, "id": 3}

<< {"jsonrpc": "2.0", "id": 3, "result": "ok"}
```

6. Reset the sensor because name and advertising duration were changed. This isn't required with firmware version 4.1 or later.

```
>> {"jsonrpc": "2.0", "method": "reboot", "id": 4}

<< {"jsonrpc": "2.0", "id": 4, "result": "ok"}
```

7. Request Disconnect.

## 4.5 Event Log

All sensor events are recorded in a non-volatile log. The log can store seven days of measurements with a temperature sample rate of 15 minutes. Each event has a timestamp and a sample ID. These values are the same as those found in the advertisements.

The log can be read in FIFO mode or LIFO mode. In FIFO mode, the oldest event is retrieved first. In LIFO mode, the last event is retrieved first.

Many systems do not require the traceability that the event log provides. When the log is full, the oldest event is overwritten.

### 4.5.1 Event Structure

The JSON format used for sensor configuration has too much overhead for large amounts of data. Therefore, the event log data is transferred with a size in bytes and a Base64 encoded array of event structures.

***Table 8: Event structure***

| Name | Size in Bytes | Description |
|---|---|---|
| timestamp | 4 | Seconds from Jan 1, 1970 |
| event data | 2 | The type of data depends on the event type.<br>Signed temperature in hundredths of degrees C, unsigned voltage in millivolts, or Magnet State. |
| type | 1 | See Record Event Types. |
| salt | 1 | A counter used by sensor to differentiate events with the same timestamp (simultaneous events). |

This is a representation of a sensor event as a C-structure. Data is formatted least significant byte first.

```
struct
{
  uint32_t timestamp;
  uint16_t data;
  uint8_t type;
  uint8_t salt;

};
```

### 4.5.2 Example

1. Connect to sensor.
2. Tell the sensor to get ready to send logs in FIFO mode.

```
>> {"jsonrpc": "2.0", "method": "prepareLog", "params": [0], "id": 1}

<< {"jsonrpc": "2.0", "id": 1, "result": 9}
```

3. Ask for 500 events. Sensor returns nine events (72/8). The sensor does not return more than 128 events per read.

```
>> {"jsonrpc": "2.0", "method": "readLog", "params": [500], "id": 2}

<< {"jsonrpc": "2.0", "id": 2, "result": [72,
"Ob/mXZIJAQA5v+ZdLgsMAXDA5l0BAAMAdsDmXXELDACRweZdLAkBAJHB5l1HCwwB0MLmXQEAAwDowuZdAQADAOjC5l0BAAMB
"]}
```

4. Acknowledge events.

```
>> {"jsonrpc": "2.0", "method": "ackLog", "params": [9], "id": 3}

<< {"jsonrpc": "2.0", "id": 3, "result": 9}
```

**Note:** The sensor allows any number of events to be acknowledged even if they haven't been read. If more events are acknowledged than were read and another read is issued, then blank entries are returned.

5. Update the time (after all events have been read).

```
>> {"jsonrpc": "2.0", "method": "setEpoch", "params": [1575404269], "id": 4}

<< {"jsonrpc": "2.0", "id": 4, "result": "ok"}
```

*Table 9: Decoded log*

| Index | Epoch | Salt | Local time | Data | Event Type |
|---|---|---|---|---|---|
| 1 | 1575403321 | 0 | 03 Dec 19 14:02:01 | 24.5 | TEMPERATURE |
| 2 | 1575403321 | 1 | 03 Dec 19 14:02:01 | 28.62 | BATTERY_GOOD |
| 3 | 1575403632 | 0 | 03 Dec 19 14:07:12 | - | MOVEMENT |
| 4 | 1575403638 | 0 | 03 Dec 19 14:07:18 | 29.29 | BATTERY_GOOD |
| 5 | 1575403921 | 0 | 03 Dec 19 14:12:01 | 23.48 | TEMPERATURE |
| 6 | 1575403921 | 1 | 03 Dec 19 14:12:01 | 28.87 | BATTERY_GOOD |
| 7 | 1575404240 | 0 | 03 Dec 19 14:17:20 | - | MOVEMENT |
| 8 | 1575404264 | 0 | 03 Dec 19 14:17:44 | - | MOVEMENT |
| 9 | 1575404264 | 1 | 03 Dec 19 14:17:44 | - | MOVEMENT |

# 5 Mobile Application

## 5.1 Overview

The **Sentrius™ BT510** mobile application allows a user to configure a device, troubleshoot a device, see real-time sensor data, and update firmware.

## 5.2 Using the Sentrius™ BT510 Sensor Mobile App on Device

To use the **Sentrius™ BT510** mobile application, follow these steps:

1. From the applicable app store (Apple or Android) search for and install the **Sentrius™ BT510** Sensor mobile application on your device.
2. To connect to the **Sentrius BT510** sensor, press and hold the button in the center of the Ezurio logo for three seconds (#3 on Figure 3).


*Figure 3: Front side of the Sentrius™ BT510 sensor*

The sensor blinks the green LED, begins advertising, and becomes connectable.

3. Tap the search icon to discover the sensor(s) within range of the mobile device (Figure 4).

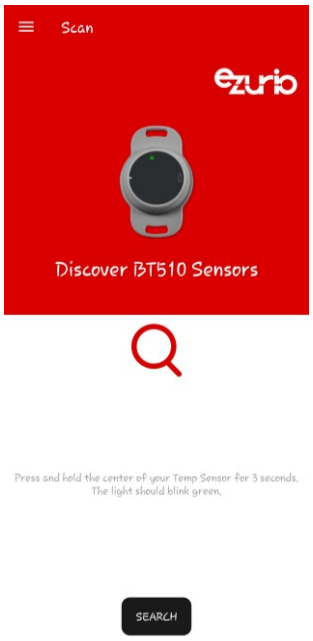



***Figure 4: Scan for new sensors***

4. Select the applicable device. The sensor name defaults to BT510 when searching for the sensor. The BLE ID is printed on the label located on the back of the sensor (Figure 5). The BLE ID matches the number of the applicable sensor located in the search results Discover screen (Figure 6).

**Note:** The number displayed in the search results is the same number as shown on the device label just in a different format. For example, from the following, the BLE ID **E8B284354832** on the label matches **BT510 – EF-30-3C-57-8B-8C** in the search results.


***Figure 5: BLE ID location***


***Figure 6: Device search results***

Once you select the desired sensor and if this is the first time connecting to it, you need to pair the selected device.

5. When prompted *Pair with BT510?*, select **Pair**. If you select **Cancel**, the application disconnects from the sensor.

   The default pairing key is *123456* (Figure 7).

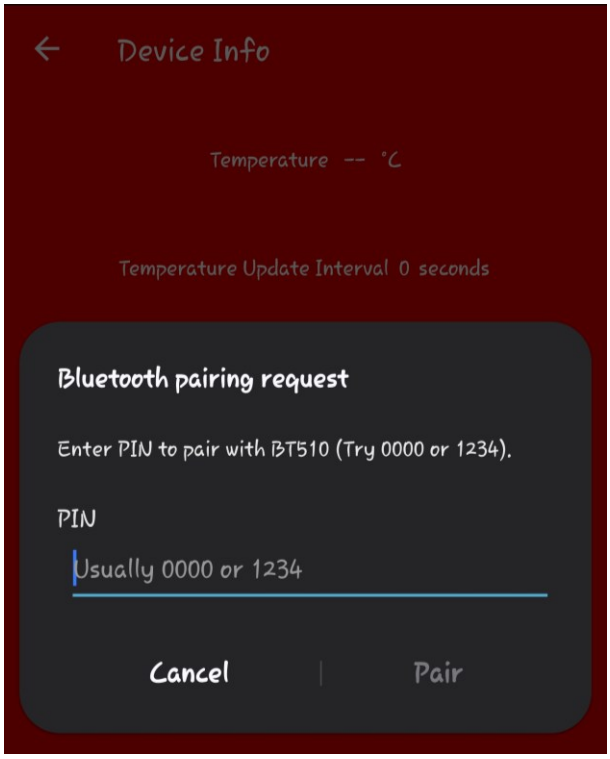



Figure 7: Pair window

Once paired, the Main screen of the mobile application displays. This screen contains a graph of the temperature over time. It also indicates at what interval the temperature value is updated by the sensor (Figure 8).

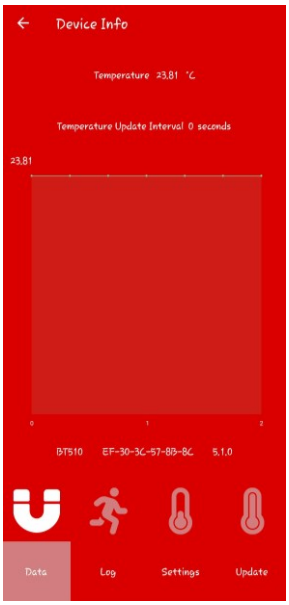



***Figure 8: Main screen***

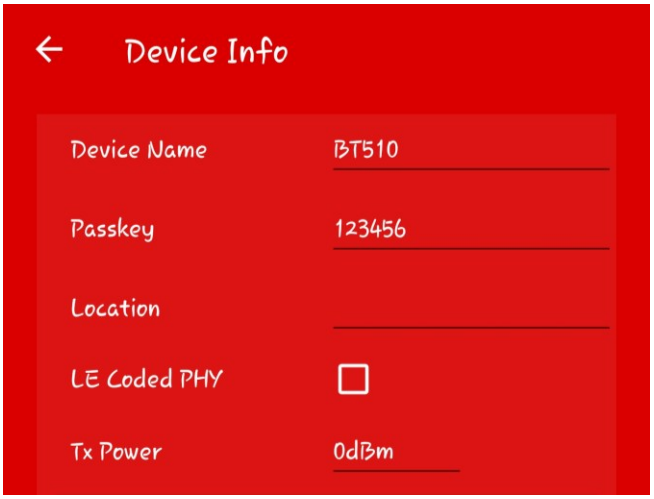
| | |
|---|---|
|  | Lights up when there is change in the magnetic switch. |
|  | Lights up when the configured *G Threshold* alarm value is reached. |
|  | Lights up when the temperature of the sensor reaches one of the configured low temperature alarms. |
|  | Lights up when the temperature of the sensor reaches one of the configured high temperature alarms. |

## 5.3 Device Configuration

The following section describes how to configure various aspects of the device. To access the configuration option, select **Settings** located near the bottom of the main screen (Figure 9).


***Figure 9: Access Settings***

### 5.3.1 Sensor Configuration




**Device Name –** Used to assign a user-friendly name to the sensor

**Passkey** – A unique number assigned to the device to help protect the configuration values from being changed without permission. This passkey is used when pairing a sensor to a new mobile device

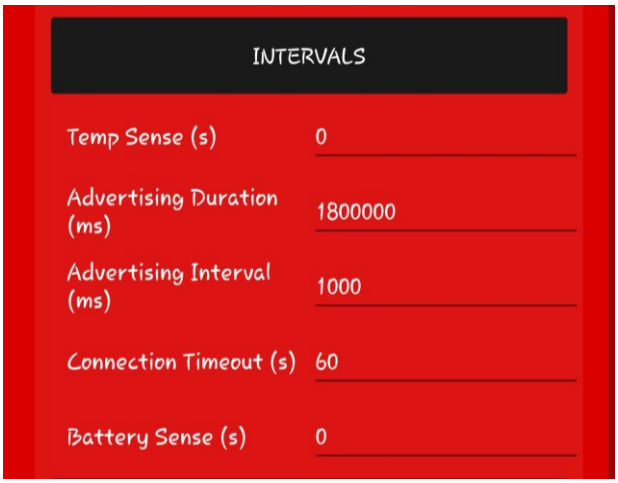
**Location** - Can be a given a name to help identify the area where the sensor is being stored.

**LE Coded PHY** – When selected the device will switch to transmitting a Coded PHY extended advertisement. The mobile app may not be able to reconnect to the BT510.

***NOTE: When selecting this option, the only way to return to 1M PHY (normal operation) is to perform a factory reset or send a JSON command.***

Pressing the save changes button for any of these configurations except Location will cause the BT510 to reset with the new value. There is a two second delay before the BT510 reset. This might cause noticeable delay in response from the mobile app.

### 5.3.2 Interval Configuration




**Temp Sense** – The time in seconds when the sensor takes a temperature measurement.

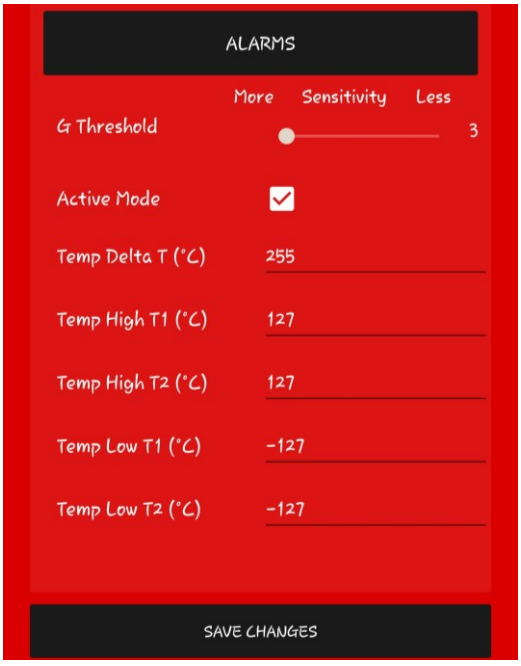
**Advertising Duration** – When set to zero the sensor advertises teach event for at least this amount and will advertise the last event indefinitely. When set to a non-zero value and stops advertising when all events have been broadcast. When set greater than 0, the duration cannot be less than four times the Advertising Interval.

**Advertising Interval** – The time between advertising packets.

**Connection Timeout** – The time in seconds that the mobile device is connected to the sensor. When set to zero, there is no timeout.

**Battery Sense** – The time in seconds when the sensor takes battery measurement.

### 5.3.3 Alarm Configuration




**G Threshold** – Sets the acceleration threshold on the sensor. The lower the number, the lower amount acceleration acted on the device is required to trigger the alarm.

**Temp Delta** – The alarm is triggered when the temperature change between intervals is greater than value written here.

**Temp High T1** – Alarm that triggers when the sensor temperature is higher than this value. It is also logged when triggered and has the high temp icon light up on the data screen.

**Temp High T2** – Behaves the same as *Temp High T1*. This is just another alarm that can be triggered if the temperature is high than this value.

**Temp Low T1** – Alarm that triggers when the sensor temperature is lower than this value. It is also logged when triggered and has low temp icon light up on the data screen.

**Temp Low T2**: This behaves the same as *Temp Low T1*. This is just another alarm that can be triggered if the temperature is lower than this value.

## 5.4 View Log

Select the Log icon located near the bottom of the screen (Figure 10). Figure 11 displays an example of log data over time.


*Figure 10: Log icon*


*Figure 11: Log data*

## 5.5 Update Firmware

Select the Update icon located near the bottom of the screen (Figure 12).


*Figure 12: Update button (Ignore Settings tab color)*

If the current firmware on the sensor is out of date, it is shown to have update package available as seen in Figure 13. Select the UPDATE button to begin the update process. It updates the bootloader first if it is out of date and then updates the firmware. Once the update is complete, a window appears that indicates success or failure (Figure 14).


*Figure 13: Available update*

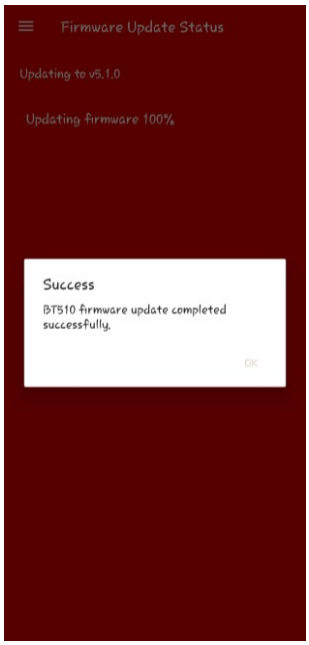



*Figure 14: Update complete*

# 6 Bluetooth SIG

The **Sentrius™** Sensor is certified by the Bluetooth® SIG as a Bluetooth v5.0, End Product. The Declaration ID (DID) is D041400.

# 7 Regulatory

**Note:** For complete regulatory information, refer to the BT510 Regulatory Information document which is also available from the BT510 product page.

The BT510 holds current certifications in the following countries (based upon being a host for the BL654):

| Country/Region | Regulatory ID |
|---|---|
| USA (FCC) | SQGBL654 |
| EU | N/A |
| Canada (ISED) | 3147A-BL654 |
| Australia | N/A |
| New Zealand | N/A |

# 8 Ordering Information

| Part Number | Description | Packaging |
|---|---|---|
| 455-00083 | SentriusTM BT510 Sensor, Single | Single (1) |
| 450-00048B | SentriusTM BT510 Sensor, Bulk Packaged | Bulk (100) |
| 455-00058 | Magnet Kit (Open/Close) for Sentrius™ BT510, Single | Single (1) |
| 455-00058B | Magnet Kit (Open/Close) for Sentrius™ BT510, Bulk Packaged | Bulk (100) |

# 9 Label Info

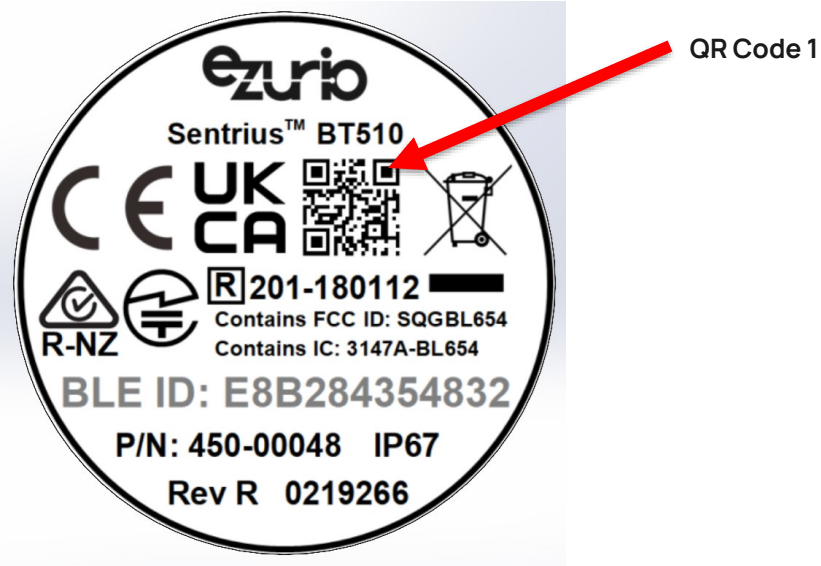



*Figure 15: BT510 back sensor label*

The QR Code contains the part number, hardware revision, date code, and BLE ID (All CAPS).

**Example Readout:** 450-00048,1,0919452,CB09AE5B8D7F

# 10 JSON Properties

| Category | Name | Type | Min | Max | Default | Units | Reset Req. | Description/ Notes |
|---|---|---|---|---|---|---|---|---|
| rw | sensorName | s | 0 | 23 | BT510 | NA | 1 | The sensor name is part of the advertisement. |
| rw | location | s | 0 | 32 | "" | NA | 0 | User configurable string that is unused by sensor. |
| rw | advertisingInterval | u | 20 | 10000 | 1000 | ms | 1 | The interval between advertising packets. |
| rw | advertisingDuration | u | 0 | 0 | 0 | ms | 1 | This specifies how long the sensor should advertise each event. Setting 0 is a special case. Internally, it will set the duration to 15x the interval. 15000 is often used for normal operation (with an advertising interval of 1000). The |

| Category | Name | Type | Min | Max | Default | Units | Reset Req. | Description/ Notes |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | duration must be at least 4 x interval. |
| rw | connectionTimeout | u | 0 | 10000 | 60 | s | 0 | 0 = none |
| rw | passkey | s | 6 | 6 | 123456 | NA | 1 | 6 digits 0 to 9 |
| rw | lock | u | 0 | 1 | 0 | NA | 0 | Reserved for future use. |
| rw | batterySenseInterval | u | 0 | 86400 | 0 | s | 0 | |
| rw | temperatureAggregationCount | u | 1 | 32 | 1 | samples | 0 | When greater than 1, the temperature samples are averaged. |
| rw | temperatureSenseInterval | u | 0 | 86400 | 0 | seconds | 0 | |
| rw | highTemperatureAlarmThreshold1 | i | -128 | 127 | 127 | °C | 0 | Range is larger than actual operating range. |
| rw | highTemperatureAlarmThreshold2 | i | -128 | 127 | 127 | °C | 0 | threhold2 > threshold 1, threshold 1 checked first |
| rw | lowTemperatureAlarmThreshold1 | i | -128 | 127 | -127 | °C | 0 | |
| rw | lowTemperatureAlarmThreshold2 | i | -128 | 127 | -127 | °C | 0 | threshold 2 < threshold 1 |
| rw | deltaTemperatureAlarmTheshold | u | 0 | 255 | 255 | °C | 0 | |
| rw | odr | u | 0 | 9 | 5 | NA | 0 | LIS2DH12_ODR_POWERDOWN 0<br>LIS2DH12_ODR_1HZ 1<br>LIS2DH12_ODR_10HZ 2<br>LIS2DH12_ODR_25HZ 3<br>LIS2DH12_ODR_50HZ 4<br>LIS2DH12_ODR_100HZ (default/tested value) 5<br>LIS2DH12_ODR_200HZ 6<br>LIS2DH12_ODR_400HZ 7<br>LIS2DH12_ODR_1620HZ 8<br>LIS2DH12_ODR_1344_5376HZ 9 |
| rw | scale | u | 0 | 3 | 2 | NA | 0 | LIS2DH12_SCALE_2G 0<br>LIS2DH12_SCALE_4G 1<br>LIS2DH12_SCALE_8G 2<br>LIS2DH12_SCALE_16G 3 |
| rw | activationThreshold | u | 0 | 127 | 8 | NA | 0 | With odr 5 and scale 2 |
| rw | returnToSleepDuration | u | 0 | 127 | 6 | NA | 0 | |

| Category | Name | Type | Min | Max | Default | Units | Reset Req. | Description/ Notes |
|---|---|---|---|---|---|---|---|---|
| ro | tempCc | i | -128000 | 127000 | NA | Hundredths of a °C | 0 | |
| ro | batteryVoltageMv | u | 0 | 0 | NA | Milli-volts | 0 | |
| ro | hwVersion | s | 1 | 1 | NA | NA | 0 | *major* |
| ro | firmwareVersion | s | 0 | 11 | NA | NA | 0 | *major.minor.revision* |
| ro | resetReason | s | 0 | 8 | NA | NA | 0 | For development |
| rw | scratchpad1 | u | 0 | 0 | NA | NA | 0 | For development.<br>This can be used to test read/write operation without affecting sensor operation. |
| rw | scratchpad2 | u | 0 | 0 | NA | NA | 0 | For development |
| rw | scratchpad3 | u | 0 | 0 | NA | NA | 0 | For development |
| ro | bluetoothAddress | s | 12 | 12 | NA | NA | 0 | |
| ro | mtu | u | 20 | 244 | 20 | bytes | 0 | For development.<br>Maximum Transmission Unit for Bluetooth Notifications and Write without response |
| ro | accelerometerSelfTestStatus | u | 0 | 4 | NA | NA | 0 | 0 Reserved<br>1 Unable to communicate with sensor (possible I2C issue)<br>2 Self-test pass<br>3 Self-test fail |
| ro | flags | u | 0 | 0 | NA | NA | 0 | Bitmask is the same as what is found<br>in the advertisement. |
| ro | resetCount | u | 0 | 0 | NA | NA | 0 | |
| rw | useCodedPhy | u | 0 | 1 | NA | NA | 1 | When 1 the coded PHY is used as the primary and secondary PHY to send connectable but non-scannable extended advertisements. |
| rw | txPower | i | -40 | 8 | 0 | dBm | 0 | -40 dBm, -20 dBm, -16 dBm, -12 dBm,<br>-8 dBm, -4 dBm, 0 dBm, +2 dBm, +3 dBm,<br>+4 dBm, +5 dBm, +6 dBm, +7 dBm, +8 dBm |
| rw | networkId | u | 0 | 65535 | 0 | NA | 0 | The Network ID is part of the advertisement.<br>It can be used to put sensors in a group that is then filtered by Network ID on a gateway. |

| Category | Name | Type | Min | Max | Default | Units | Reset Req. | Description/ Notes |
|---|---|---|---|---|---|---|---|---|
| rw | configVersion | u | 0 | 255 | 0 | NA | 0 | The version can be used to help synchronize changes made by a phone application and a gateway. This value is unused by the sensor. |
| ro | bootloaderVersion | s | 0 | 11 | NA | NA | 0 | *major.minor.revision* |

# 11 Frequently Asked Questions

https://www.ezurio.com/support/faqs

## What is the command to enter the bootloader (Nordic Secure DFU)?

```
{"jsonrpc": "2.0", "method": "reboot", "params": [1], "id": 1}
```

## Where can I find more examples?

There are Python examples that use the BL654 USB dongle that can be found at https://github.com/LairdCP/BT510-Python.

## What should the advertisement duration and interval be set to?

The interval and duration are system parameters and can be tuned to each use case.

- They affect response time.
- The interval affects battery life. Advertising more often will reduce the battery life.
- They should be set to values that prevent a gateway from missing events.
- If there are many Bluetooth devices, then the probability of RF collisions increases especially if some devices are advertising at sub 250 ms intervals.

The processing power of the gateway is also a factor. For example, the MG100 not only scans for the BT510 advertisements, but it operates the cellular modem, advertises, makes connections to configure other BT510 sensors, and reads sensor data from the BL654 Sensor.

Unfortunately, it is not possible to guarantee that every event will be seen by the gateway. If this is a system requirement, then the gateway must read the sensor event logs when an event is missed.

The duration controls how long each event is advertised. If the sensor is only doing one thing, for example, measuring temperature, then the duration is not that important because the sensor advertises the last event forever. However, if the sensor is measuring temperature and there are alarms enabled, then the duration is important. The sensor will measure temperature and also determine if any alarms have been set. If the low alarm has been set, then the sensor will advertise the temperature and then it will advertise the alarm. If the duration is long enough, the gateway can detect both events.

| Duration | Interval | Notes |
|---|---|---|
| 15000 | 1000 | Default in version 5.0. Balance of good battery life, response time, and number of advertisements when there are concurrent events. |
| 1000 | 100 | Fast response time. If there are multiple concurrent events, then duration may be too short for gateway. Increasing interval reduces battery life. |
| 2000 | 100 | Duration of greater than 1 second ensures that gateway should see event. Assuming the gateway takes less than 500 milliseconds to send event to cloud, this can provide ~1 second response time. |
| 1800000 | 1000 | Default in some versions prior to 5.0 (for production reasons). The response time is 30 minutes. If measurement rates are set faster than 30 minutes or the sensor is setup for movement/magnet detection, then events may be discarded (but still logged to non-volatile memory). The sensor can queue up 16 events to advertise. |
| 15000 | 5000 | Invalid. Duration must be greater than 3 x interval. |
| 20000 | 5000 | If there are concurrent events, then some events will only be advertised 4 times. Intervals greater than 1 second make the time for a connection appear slow for humans. |

## Why don't the advertisements match the event log? Why are events missing?

As described in Section 4.1, the sensor has an advertisement queue. This queue contains 16 entries. This allows the sensor to store concurrent events and then periodically broadcast the next event. When events occur faster than the advertising duration and the queue is full, they won't be added to the advertisement queue, but they will still be stored in the event log.

For versions prior to 5, if the advertising duration is set to 0 the sensor always bypasses the advertisement queue and immediately advertises any event. Depending on the sensor configuration, this can result in many un-advertised events.

For versions later than 5, if the advertising duration is set to 0, the duration used by the sensor is 15 * advertising interval. This was done to prevent events from not being advertised.

For any version, if the sample rate (for example, temperature interval) is set to faster than the duration, the event queue will have to discard items. A temperature sample rate of 10 seconds and an advertising duration of 15 seconds will result in an advertisement queue that is always full and un-advertised events. Once the queue is full, advertised events are those that occurred about 4 minutes in the past.

## What are the recommended settings for cold operation?

When operating below freezing it is not recommended to advertise indefinitely because it doesn't allow the battery to recover. It is also recommended to keep Bluetooth connections as short as possible. If the system doesn't require connections, then they should be avoided.

# 12 Additional Information

Please contact your local sales representative or our support team for further assistance:

| | |
|---|---|
| **Headquarters** | Ezurio<br>50 S. Main St. Suite 1100<br>Akron, OH 44308 USA |
| **Website** | http://www.ezurio.com |
| **Technical Support** | http://www.ezurio.com/resources/support |
| **Sales Contact** | http://www.ezurio.com/contact |

**Note:** Information contained in this document is subject to change.

Ezurio's products are subject to standard Terms & Conditions.