Aqua-Scope Gen2
Communication Interfaces

Version: November 8th, 2023
Author: C.Paetz (cp@aqua-scope.com)
Translated using Chat GPT Version 3.5

All Aqua-Scope devices with a WLAN interface (AQS Monitor, WIFI Rain Level Sensor, Ultrasonic Clamp, etc.) support multiple communication protocols that can be used individually or in parallel. These various communication protocols all offer the possibility to enable/disable and configure each communication protocol separately.

The most convenient way to enable/disable communication channels is to use the Aqua-Scope App. In the "Info" section of the app, you can find general information about the app. Scroll all the way down to the "Debug" button. Here, you will find information about the device, configuration parameters, the ability to directly set parameters, and a list of switches for enabling and disabling individual communication channels.

When enabling the JSON or MQTT channels, additional configuration details are required for these channels. Please note that JSON, MQTT, and AQS Cloud require a WIFI IP connection. If this connection is not active, these communication channels will also be deactivated, or the WIFI switch will be automatically activated when one of the mentioned channels is active. When enabling the JSON or MQTT channels, additional configuration details are required for these channels. Please note that JSON, MQTT, and AQS Cloud require a WLAN IP connection. If this connection is not active, these communication channels will also be deactivated, or the WLAN switch will be automatically activated when one of the mentioned channels is active.

Aqua-Scope Cloud Interface

By default, the devices communicate directly with the Aqua-Scope Cloud via an IP connection. The data is encoded as a JSON object and then encrypted using the XXTEA encryption algorithm with a private key stored in the device. The encrypted data is transmitted as ASCII hexadecimal values in the POST field of an HTTP request. In the data field of the HTTP request, the downlink data is also sent as a JSON object and encrypted using XXTEA.

The following downlink commands are understood by the devices as JSON objects:

General JSON Downlink Objects

`'conf':{a:b,c:d}`
Setting configuration parameters. The communication parameters are device-specific but are always transmitted via the JSON object 'conf'. A maximum of 32 parameters can be changed simultaneously. The parameters are modified in the order they are specified in the object. 'a' and 'c' represent the parameter number, while 'b' and 'd' are the respective values in decimal format. Parameters are stored internally in the device as 32-bit values. A set of general configuration parameters you find in this document in section “General Configuration Parameters”. Further device - specific configuration parameters are provided in the device manuals.

`'system': X`
System commands are:

- x=1 Reboot the system.
- x=2 Reset the device to its factory default state.
- x=3 Mute the device, causing it to stop communicating until the next power-on.

`'shell': 'aaa'`
Execute a command on the serial shell. The commands are described in the BLE interface. Note that if a command requires a space, use the underscore character ('_').

`'ota': 'XXX'`
Firmware Update Over the Air. XXX represents a transaction number used to request the required firmware from the OTA server.

`'webs': { 'enable': '1', 'url': 'https://io.aqua-scope.com/be3/test.php', 'token': '12345', 'raw': 1, 'xxtea': 1 }`
The 'webs' object configures the use of a JSON Web Service. When enabled (enable = 1), all JSON reports are encoded and transmitted in the POST field of an HTTP request. Optionally, a private token can be provided, which is appended to the URL as a GET parameter. With 'raw' set to 1, detailed raw data is also sent to this service when the device collects such data. The 'xxtea' switch allows the JSON object to be encrypted with the device's private key. More information on this is available in the JSON Web Services section.

`'clear': X`
Clears an alarm of type X in the device. Using X = 0 will clear all alarms in the device. The alarm will also be reset even if the cause of the alarm, such as a temperature drop, has not been resolved. When an alarm is cleared with this command, the alarm will not be triggered again until the next reboot (via command or power-on).

`'mqtt':{'debug':'1','enable':'1','server':'aaa.bbb.com', 'port':789,'login':'abcdef','password':'abcdef'}`
Configures the use of an MQTT service. When enabled (enable=1), all reports and alarms from the device are sent to the MQTT service specified as 'server' in parallel with other communication channels, using the IP port 'port'. Depending on the configuration of the MQTT service, an optional user login and password may need to be provided. The values can be subscribed to on the MQTT broker via the string '/AQS-XXXXXX', where XXXXXX is the 6-digit ID of the device.

`'list': 0`
Transmits all configuration parameters of the device as an array.

`{"eid":"12345678","sensor_x":XXXX,"sensor_y":YYYY, …}`
The status report of a device is typically sent every 15 minutes and contains the names and current values of device-specific sensors. A status report is also sent for battery-operated devices. The interval (time between two reports) is defined in configuration parameter 29 (default 900 seconds). This does not apply to Aqua-Scope devices with a water pressure sensor (AQS*, PRE*, AQE*, ...) where the status is sent along with the raw data depending on the chosen sampling interval. The sensor types are provided in the device manual.

JSON-Downlink Objects specific to the Aqua-Scope Monitor

The Aqua-Scope Monitor and Aqua-Scope Monitor Plus extend the general set of command objects:

`'cmd':{a:b,c:d}`
Execution of a device-specific command: Multiple commands with numbers and a parameter can be executed. If a specific command does not require a parameter, 0 should be specified.

The following commands are accepted by the Aqua-Scope Monitor:

- 1:0 – Start PipeCheck
- 2:0 - Abort Pipe-Checks
- 3:x – Switch Device Type: x=1-PREWIE02, x=2-PRELWE02, x=3-AQXWIE02, x=4-AQSLWE02, x=5-AQSWIE02, x=6-AQSLWE02
- 4:0 – Reset Detection Algo back to factory default, calibration restarts
- 4:1 – Activate T-Algorithm with standard values
- 4:2 – Activate M-Algorithm with standard values

`'sub': {X: {'c': cmd,'d': channel,'I': deveui,'k': key,'t': type},Y: {// ...}}`
This configures sub-devices of an Aqua-Scope Monitor, whose functionality can only be used through the Aqua-Scope Monitor. Communication between these sub-devices and the Aqua-Scope main device occurs through a specially secured private LoRa Point-2-Point connection. Various commands are supported and are processed in the order they are specified in the JSON object. The command number X is not considered, but it's used to separate different commands.

- Cmd = 1: Registers a new device with the 16-digit ID 'deveui' and the 16-digit private key 'key' on one of the 10 available 'channels' of the Aqua-Scope Monitor. If a device is already registered for a specific channel, this entry will be overwritten. Additionally, a device type (positive number) must be specified when adding a device. For

actuator motors, these types can be 1 for ball valve motors or 3 for angle seat valve motors.
- Cmd = 2: Removes a device with the specified 'deveui' from the communication 'channel'. The device is deleted only if a device with the ID 'deveui' has been previously registered for the 'channel'.
- Cmd = 3: Changes the device type of the device in the 'channel' to the new device type 'type'. The reason for this command is evident from the 'valve' or 'svalve' command.

`'valve': X`
Sets a motor registered with the Aqua-Scope Monitor to the value X, which can be either 0 (closed) or > 0 (open). If more than one motor (device types 1 or 3) is registered, these motors are closed or opened together. If this is not desired, the type of a motor can be changed to a value other than 1 or 3. This motor must then be controlled directly using the 'svalve' command.

`'svalve': {c: X,s: Y}`
Sends a command to open or close a motor with a value Y to a sub-device in communication channel X, regardless of its device type. This command is needed when a motor is separated from the common control of all motors with the command 3. If the device in communication channel X is battery-operated and in deep sleep, this command will be executed when the device wakes up next. Pressing a button on the sub-device will wake it up.

`'sconf': {c: X,p: Y,v: Z}`
Sets a configuration parameter 'Y' of a sub-device in communication channel 'X' to the value 'Z'. If the device in communication channel X is battery-operated and in deep sleep, this command will be executed when the device wakes up next. Pressing a button on the sub-device will wake it up.

General Uplink-Objects specific to the Aqua-Scope Monitor

`{"eid":12345678,"list":[1, 2, 3, 4, …, 30]}`
Sends an array with all configuration parameters of a device in response to the downlink command 'list'.

`{"eid":"12345678","uptime":XXXX,"state":YY}`
The heartbeat of a device transmits the uptime in seconds plus a device-specific status. If a meaningful status cannot be transmitted for a device, this object is omitted. A heartbeat is only sent for devices that are not battery-operated. The interval (time between two heartbeats) is defined in configuration parameter 30 (default 20 seconds).

`{"eid": "12345678", "arr": XXXXX}`
If available on the device, this object transmits the raw data from the main sensor as a continuous string of hexadecimal characters of length 4. For

example, "arr": "035403560350..." for 0x0354 = 852, 0x0356 = 854, 0x30350 = 848.

General Uplink-Objects

Here is a breakdown of the provided JSON objects:

```
{"eid":"12345678","flow":{'event':'1|2', 'dur': XXX, 'cons': YYYY, 'reason': ZZZ } }
```

This is a notification for the start and end of detected water consumption. 'event' = '1' indicates the start of a water consumption event with information about the reason for the event, and 'event' = '2' marks the end. It includes the event duration ('t'), the amount of water consumed ('l'), and, for debugging purposes, a possible reason for the termination of water flow. The duration is in seconds, and the amount is in milliliters.

```
{ "eid": "12345678", "pc": { "state": X, "dur": YYY, "diff": Z, "elev": EE } }
```

This is a status report for a Pipe-Check. Possible status values include:

- 2: OK, no leak detected, 'dur' indicates the duration of the performed Pipe-Check in seconds.
- 3: Leak detected, 'dur' is the duration of the Pipe-Check, 'diff' is the measured pressure difference, and 'elev' is the calculated height of the detected leak above the sensor.
- 4: Pipe-Check aborted due to detected water demand.
- 5: Pipe-Check aborted due to detected hot water production.
- 6: Pipe-Check aborted due to a valve error.
- 7: Manual Pipe-Check abortion.
- 8: Pipe-Check started.

```
{"eid":"12345678","sub":{"c":X,"pay":YYY}}
```

This is raw data from a sub-device in channel 'X'. Decoding the raw data can be done on the server. The encoding corresponds to the LoRaWAN communication channel encoding. The main device will also decode the received payload to execute commands intended for the main device (e.g., alarms that lead to motor shutdown).

```
{"eid":"12345678","alarm":{"state":X,"type":Y,'val':Z,"map":AA}}
```

This alarm object notifies active (state=1) or completed (status=0) alarm states. 'type' specifies the alarm type, and depending on the alarm type, an additional value may be provided if applicable (e.g., the current temperature value for a temperature alarm). The complete alarm vector 'map' can also be transmitted, where each bit corresponds to an alarm type. The general alarm types and their corresponding bit values are described in the section “Alarm types” of this document. Further device-specific alarm types are described in the device manuals.

JSON-Web-Service

To use this communication channel, you need an HTTP server with a communication endpoint to which the device can send its data in JSON format via the POST field. The communication options are defined in the Aqua-Scope Cloud Interface description.

If encryption via XXTEA (https://en.wikipedia.org/wiki/XXTEA) is chosen, you must have a device-specific 128-bit encryption key. This key is securely embedded in the device and cannot be extracted or read. You can obtain this key directly from Aqua-Scope upon request, providing the 8-byte ID. The 8 bytes of the ID follow the format specified by Aqua-Scope.

AA BB BB BB CC CC CC CC

- AA = ID of the device (0x06 = RANWIE01, 0x0a = USSWIE01,0x0b = AQS family)
- BB = 24 Bit long serial number padded with zeros
- CC = random 4 Byte

When no encryption is selected, data transmission occurs as an HTML POST with a content type of 'application/json.' With encryption, 'application/raw' is used, and the JSON object is encrypted with XXTEA and then encoded as HEX ASCII.

The transmitted JSON object corresponds to the Aqua-Scope Cloud Interface description.

As a response or command to the device, a downlink object can be sent in the HTTP text, the format of which also complies with the Aqua-Scope Cloud Interface.

The transmission of sensor raw data (using the 'arr' object) must be explicitly enabled, for example, in the app. This data is generally not encrypted.

The following small example demonstrates the reception of a JSON object from the Aqua-Scope device, exemplified as PHP code."

If you have any further questions or need assistance with anything specific, feel free to ask.

```
<?php
require_once __DIR__ . 'xxtea-php.php'; // from
https://github.com/xxtea/xxtea-php

#define EID 060000001
#define KEY 060000001234567806000000012345678
```

```
error_reporting(0);
$json  = json_decode(file_get_contents('php://input'), true);
if(!$json) {
      $key = hex2bin(KEY);
      $jd =
xxtea_decrypt(base64_decode(file_get_contents('php://input')),$key);
      $json  = json_decode(jd, true);
      if(!$json) die(”wrong key or wrong id”);
}
var_dump($json);
?>
```

MQTT Service

All data (except for the raw data from the main sensor) can be sent in parallel to both the JSON web service and the Aqua-Scope Cloud to an MQTT broker. To do this, the server, port, and optionally an access login and password are provided. To retrieve data, you must subscribe to:

/AQS-12345678/*
/AQS-12345678/alarm

With '12345678' as the 8-digit device ID. It is also possible to send downlink commands as JSON objects via the MQTT service. The notation of these commands is identical to those of the Aqua-Scope Cloud.

Bluetooth Low Energy

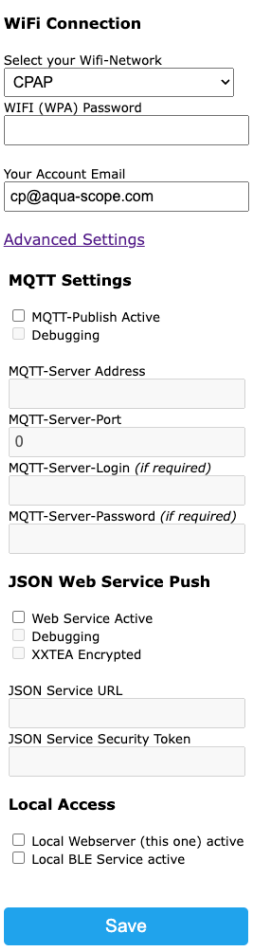
The Bluetooth interface of Aqua-Scope WLAN devices is active by default and is used for the initial configuration of WLAN through the Aqua-Scope app. Once the WLAN interface is successfully configured, the Bluetooth interface is deactivated. However, it can be reactivated from any communication channel. Please note that reactivating the Bluetooth interface on battery-operated devices will block their sleep mode and quickly deplete the battery.

The Bluetooth interface implements the UART profile in BLE 5. It can be used with all serial Bluetooth applications. It's even more convenient when using a PC or an Android mobile phone with Bluetooth capabilities. You can open the Chrome browser, navigate to the website 'https://ble.aqua-scope.com,' and connect to the Bluetooth device 'AQS-XXXXXX' (where XX is the device's ID).

Once the Bluetooth connection is established, you have direct access to the internal serial console of the device. You can view all debug messages and execute commands, seeing their results on the console. The following commands are supported:

- 'mal XXX': Set the email address of the AQS account for the device.
- 'sid XXXX': Set the SSID for WLAN access.
- 'wpa XXX': Set the WPA key for WLAN access.
- 'wce XXX': Set the URL of the JSON service.
- 'wto XXX': Set the optional token of the JSON service.
- 'mse XXX': Set the server name of the MQTT server.
- 'mpo XXX': Set the IP port of the MQTT server.
- 'mpa XXX': Set the password for the MQTT server.
- 'mus XXX': Set the username for the MQTT server.
- 'configget XX': Display the value of configuration register XX.
- 'configset XX YYYY': Set configuration parameter XX to the 32-bit value YYYY.
- 'configlist': Display a list of all configuration parameters.
- 'resetc': Restart the system.
- 'alarm': Display alarm registers.
- 'sleep': Enter sleep mode.
- 'status': Display the status of the main processor.
- 'factory XXXX': Factory Reset when XXX=2010 is entered.
- 'mon X': X=0,1,2, Activate ULP monitoring. 0 = all debug messages off, 1 = commands from ULP to the main processor are displayed, 2 = a status report of the ULP is displayed every second.
- 'ubilink c, v1, v2, v3, v4': Direct control of a sub-device in channel 'x' with bytes v1...v4. The commands correspond to LoRaWAN devices. You can specify 0,1,2,3, or 4 bytes.
- 'sensors': Display the list of sub-devices with keys and status.
- 'buffers': Display the output buffers of the respective channels to sub-devices.

- 'ulpmode': Allows switching the ULP processor to different modes.
- 'ulpstatus': Display the status word of the ULP processor.



Local Web Server

The local web server of Aqua-Scope WLAN devices is active by default on Port 80 (HTTP service) and can be used for the initial configuration of WLAN, among other functions. Once the WLAN interface is successfully configured, the web server is deactivated. However, it can be reactivated from any communication channel.

The web server allows for enabling and disabling, as well as configuring other communication channels like MQTT or JSON web service. It also provides easy access to the current sensor values of the device. Accessing the web server when the device is in sleep mode is challenging as there is a very limited time available for querying or setting values.

The web server interface is self-explanatory.

LoRaWAN Interface

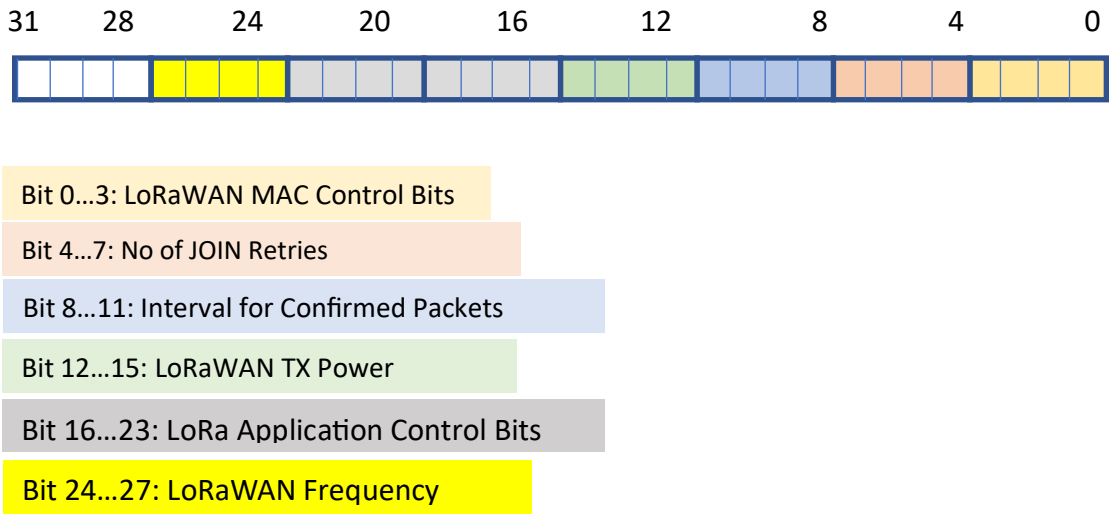
General Information: The Aqua-Scope monitor is equipped with a LoRaWAN communication system based on the Semtech SX1262 IC plus Semtech Library for **MAC version 1.0.2** with regional parameters **PHY V.1.0.2 REV B**.

Configuration

LoRaWAN communication is enabled/disabled using Bit 2 (x |= 0x4 / x &= ~0x04) of the AQS systems register as configuration parameter #1. For all SKUs with the ‘xxx**LW**xxxx’ this bit is enabled on default.

The behavior of the LoRaWAN communication I controlled by the LoRaWAN control register as config parameter #3.

This configuration parameter has 32 bits and can be changed by configuration set commands. Please note that the LoRa system can only change the bits 0...15 but no the bits 16-31. This ensures that LoRa itself cannot change the LoRa access rights or the LoRa frequency.



Bit 0...3: LoRaWAN MAC Control Bits (changes to the bits will take effect after the next reboot)

- Bit 0 (Default = 0): LoRaWAN Class A (0) versus Class C (1)
- Bit 1 (Default = 0): No ADR (0) versus ADR (1)
- Bit 2 (Default = 0): No Duty Cycle Control (0) versus Duty Cycle Control (1)
- Bit 3 (Default = 0): not used

Bit 4...7: 4 Bit integer value = number of LoRaWAN JOIN resents (Default = 3) (changes to the bit will take effect after the next reboot)

Bit 8...11: 4 Bit Integer value = confirmation Interval. The LoRaWAN system sends unconfirmed messages. Every x packets a confirmed packet is sent upstream. If this parameter is set to 0, every packet is sent with confirmation. Changes to this value will take effect immediately.

Bit 12...15: 4 Bit Integer value = LoRaWAN TX Power from 0db to 22 dB in increments of 2 dB: 0 = 0 dB, 7(default) = 14 db. Values above 11 = 22 dB are not allowed. Changes to this value will become effective after reboot.

Bit 16... Bit 23: LoRaWAN application-level control

- 0x00010000 (Bit 16): Alarm Reporting and Clearing
- 0x00020000 (Bit 17): Water Consumption Report
- 0x00040000 (Bit 18): Sensor Value Report
- 0x00080000 (Bit 19): Config Parameter Change

- 0x00100000 (Bit 20): Pipe-Check
- 0x00200000 (Bit 21): Periodic Config Reporting
- 0x00400000 (Bit 22): Fixed Payload

- 0x00800000 (Bit 23): not used

Bit 24...27: 4 Bit Integer value = frequency used. Changes to this value will become effective after reboot.

- 0: REGION_EU868 (default)
- 1: REGION_US915
- 2: REGION_AS923
- 3: REGION_AU915
- 4: REGION_CN779
- 5: REGION_IN865
- 6: REGION_KR920
- 7: REGION_RU864

JOIN Process

The LoRaWAN communication uses OTAA for joining to the network.

After power-On the devices tries to join the network with. The parameters are set in LORA Register Bit 0...3.

If the JOIN is successful, the join status is maintained by the device and is checked whenever there is a confirmed communication. This is the reason why even for battery option there is a need for confirmed communication is required from time to time.

If the JOIN status is lost due to restart of server to multiple lost packets resulting in desynchronization of packet counters the following behavior applies.

- If there was a successful JOIN, a REJOIN is attempted.
- If there is no other LoRaP2P device connected to the Aqua-Scope monitor, a REJOIN will be tried even if there was no successful JOIN before.

If a REJOIN fails again, each subsequent REJOIN is performed after an increasing period of time, starting at 15 minutes and doubling up to one day. If a JOIN has not been possible after one day, it will be retried every day if the conditions for a REJOIN are met.

Communication

In case of a successful JOIN, the following communication takes place on **FPORT = 10**:

- Status report according to the Ubilink LoRa P2P protocol specification.

- The time between regular status reports is defined in configuration register #30. Default value is 900 seconds = 15 minutes
- Response to communication requests with downlink commands. Note that the response will arrive with the next regular uplink command that does not violate the duty cycle rule.
- Unsolicited alarm messages according to the Ubilink LoRa P2P protocol specification.

When bit 22 of the LoRa control register is set, the status message is sent as a fixed size, fixed format 8 or 10 byte long message with 5 * 16 bit variables of the following format

A A B B T T P P C C

AA = 16 Bit Vector of alarm bits. The different bits represent the different alarm types. Alarm Type 3 = Bit 3 of the Alarm Vector; for alarm types please refer to the device manual
BB = 16 Bit Value of Batter draw in mAh
TT = 16 Bit Value of temperature of 1/10 Degree Celsius
PP = 16 Bit Value of Water Pressure in mBar
CC = 16 Bit Value of Water consumption, only sent from Aqua-Scope Monitors but not from Pressure sensors

Every unconfirmed packet is assumed as being delivered and will not be resent. If a confirmed communication fails the send buffer remains and the very same value will be sent again once communication has been reestablished.

Otherwise the following uplink-commands are supported:

- 0x0b 0xss 0xtt 0xvvvv: Report Alarm with state xx (1 = on, 0 = off) of type ‘t’ with optional 16 bit value vvvv
- 0x06 0xtt 0xvvvvv: Report of Sensor value of type ‘ss’ as 16 Bit Value ‘vvvv’
- 0x04 0xpp 0xvvvv: Report of Configuration Value pp as 16 Bit Value ‘vvvv’
- 0x07 0xss 0xdddd 0xeeee: Report of Pipe-Check. ‘’ss’ encodes the status. For detailed info about Pipe-Check Status please refer to chapter “AQS Coud”. 0xdddd is the optional 16 bit value of the pressure difference, 0xeeee is the optional value of the elevation of a detected leak above the position of the sensor
- 0x12 0xvv 0xbbbb: Battery report with 8 Bit Value vv as battery voltage with 1/10 Volt and Battery consumption as 16 bit value 0xbbbb in mAh.
- 0x03 0xhh 0xcccc: Report Hardware version 0xhh and hardware capability bitmap as 16 Bit value 0xcccc:
- 0x0a 0xff ff ff ff: Reports the firmware version as 32 bit value 0xffffffff.

Mains versus Battery Operation

The default value of the LoRaWAN Register is optimal for mains powered devices under normal conditions.

In case the device recognized battery operation certain LoRaWAN values are changed automatically to preserve battery power and ignoring all previous changes. In case the mains power comes back the changes will remain avoiding any flip-flopping of the values.

- Number of Join Retries is set to 3
- Confirmation Interval is set to 10
- LoRaWAN class is set to Class A

Optimizing for harsh wireless conditions

LoRaWAN offers several ways to improve communication reliability. Please note that reliability and low (battery) power consumption are mutually exclusive goals!

- Increase the number of join attempts
- Use only confirmed communication
- Increase TX power up to 22 dB
- Enable adaptive data rate

Using LoRaWAN and LoRaP2P in parallel

It is possible to use LoRaWAN and LoRaP2P in parallel. However, there are certain restrictions and rules that must be followed:

1. When setting up the LoRaWAN device in the LoRaWAN network server, you must use SF9 for RX2 window. In TTN this is called 'Europe 863-870 MHz (SF9 for RX2 -recommended)'.
2. ADR must be disabled (lora_register &= ~0x02)
3. Device must be joined as CLASS_C device.

LoRaWAN Downlink Commands

- 0x07 0xss: Start (ss=0x01) or stop (ss=0x02) the Pipe-Check process.
- 0x01 0xss: System commands:
    - ss=1: reboot the device,
    - ss=2: factory reset,
    - ss=3: endless loop to suppress further communication, ended by power-on.
- 0x0b 0xss: Reset an alarm of type ss. ss=0 resets all alarms.
- 0x04 0xss 0xyyyy: Set configuration parameter ss to the value yyyy (yyyy is a 16-bit value).

- 0x41 0xss: Query configuration value ss. A corresponding Uplink command is sent as a response.
- 0x06 0xss: Query sensor type ss. A corresponding Uplink command is sent as a response.

General Configuration Parameters

The following configuration parameters apply to all Aqua-Scope devices:

- Parameter 1 – System Register: This is a bitmap with various switches.
  - Bit 1 (0x00001): If this bit is set, the device goes into sleep mode.
  - Bit 6 (0x): The local buzzer, if present, is active (1) or inactive (0)
  - Bit 7 (0x): =0, local LED is turned off even when the device is not sleeping; =1, local LED indicates operational states.
- Parameter 3 – LoRa: This register configures the behavior of the LoRaWAN subsystem (if present as hardware). Further information is provided in the 'LoRaWAN' section.
- Parameter 11 – Lower temperature threshold: When the measured temperature falls below this threshold, a temperature alarm is triggered.
- Parameter 16 – Upper temperature threshold: When the measured temperature exceeds this threshold, a temperature alarm is triggered.
- Parameter 19 – Alarm Mask: This bitmap defines which types of alarms are active.
- Parameter 27 – Report Multiplier: When the sensor in battery operation doesn't detect changes in sensor values, it won't send a report after each interval (Parameter 29) but only every X reporting intervals. X is the value of configuration parameter 27. This increases battery life.
- Parameter 28 – Communication Channels: This register defines which communication channels in the device are active as a bitmap:
  - XXTEA = 0x001
  - JSONSERVICE = 0x002
  - MQTTSERVER = 0x004
  - BLUETOOTH = 0x008
  - HTTPSERVER = 0x010
  - WIFIENABLE = 0x080
  - LORAWAN = 0x100
  - LORAP2P = 0x200
  - JSONRAW = 0x400
  - AQSCLOUD = 0x800
- Parameter 29 – Reporting Interval in seconds
- Parameter 30 – Heartbeat Interval in seconds
- Parameter 31 – Device Type: Changing this value without correspondingly changing other settings can lead to the device becoming unusable.

General Alarm types

The alarms reported upstreams on all communication channels refer to an integer value as alarm type. The following alarm types are supported on all Aqua-Scope Devices. All alarm reports contain a status value (1 = on, 0 = off, 2 = warning), an alarm type and a value associated with. This alarm. For some alarm there is no meaningful value available an 0 is transmitted.

- Type = 2 (mask = 0x0002): **Temperature Alarm.** The measures temperature value has exceeded either the high or the low temperature threshold defined in configuration parameter 11 or 16. The value contains the current temperature value in 1/10 of degree Celsius
- Type = 11 (0x0400): **Wireless RSSI value too low**. The value contains the RSSI value of the last communication. Receiving this alarm means that the communication is still possible but there is danger of loosing connectivity.
- Type = 12 (0x0800): **Battery Low**. The value is not set.
- Type = 13 (0x1000): **Lost Mains Power**. The value is not set.

Beside these alarm types there are device-specific devices types. Please refer to the device manual for further details on these specific device types.