

# Senseceiver 4-20mA

## Product Manual

Rev .04



## Contents

|            |                                                                     |           |
|------------|---------------------------------------------------------------------|-----------|
| <b>1</b>   | <b><i>*Document History:</i></b> .....                              | <b>3</b>  |
| <b>1</b>   | <b><i>Product Description</i></b> .....                             | <b>4</b>  |
| <b>2</b>   | <b><i>Device Variants</i></b> .....                                 | <b>4</b>  |
| <b>3</b>   | <b><i>Key Benefits</i></b> .....                                    | <b>4</b>  |
| <b>4</b>   | <b><i>Connectors and Interfaces</i></b> .....                       | <b>4</b>  |
| <b>5</b>   | <b><i>IO Connector Pinout</i></b> .....                             | <b>5</b>  |
| <b>6</b>   | <b><i>Connecting a 4-20mA Sensor</i></b> .....                      | <b>6</b>  |
| <b>7</b>   | <b><i>Connection to Cloud</i></b> .....                             | <b>6</b>  |
| <b>7.1</b> | <b><i>Preparing the AWS IoT Cloud</i></b> .....                     | <b>6</b>  |
| <b>7.2</b> | <b><i>Preparing the Cumulocity Cloud</i></b> .....                  | <b>10</b> |
| <b>7.3</b> | <b><i>Preparing the Microsoft Azure IoT Hub Cloud:</i></b> .....    | <b>11</b> |
| <b>8</b>   | <b><i>Sensor Threshold Comparison and Output Trigger:</i></b> ..... | <b>13</b> |
| <b>9</b>   | <b><i>Sensor Reading:</i></b> .....                                 | <b>13</b> |
| <b>10</b>  | <b><i>Configuration tool</i></b> .....                              | <b>14</b> |
| <b>11</b>  | <b><i>LEDs behavior:</i></b> .....                                  | <b>15</b> |
| <b>12</b>  | <b><i>MQTT Message Structure</i></b> .....                          | <b>15</b> |

1 **\*Document History:**

| Revision | Date       | Edited By | Changes                                                     |
|----------|------------|-----------|-------------------------------------------------------------|
| REV.1    | 2020.06.09 | TP        | Initial release                                             |
| REV.2    | 2020.08.19 | MJ        | Added configuration tool.<br>Completed missing sections.    |
| REV.3    | 2020.08.19 | MJ        | Added description of the configuration tool                 |
| REV.4    | 2020.09.08 | MJ        | Added description of the configuration in the cloud section |
|          |            |           |                                                             |
|          |            |           |                                                             |
|          |            |           |                                                             |
|          |            |           |                                                             |
|          |            |           |                                                             |

## 1 Product Description

The Senseceiver 4-20mA is an embedded IoT/M2M hardware which supports many IoT Cloud Platforms such as AWS IoT, Azure IoT Hub, and Cumulocity IoT. Data is sent to the dedicated Cloud over Cellular networks (2G, LTE-M, and LTE-CAT1).

## 2 Device Variants

The Senseceiver 4-20mA is available in two variants with the following order codes:

|  |                       |
|--|-----------------------|
|  | Supports 2G and CATM1 |
|  | Supports LTE-CAT1     |

## 3 Key Benefits

Connect up to 4 sensors with 4-20mA interfaces to the device, and see the data in the cloud. The Senseceiver 4-20mA can be configured to calculate the actual value of the sensor based on the current in mA reading by configuring the parameters a, b, and c in the following equation:

$$\text{Sensor} = \frac{a}{b} * \text{Current\_mA} + c$$

In the case where you are using a distance sensor or a temperature sensor, you can send the actual distance or temperature being measured instead of the mA current being measured.

The device sends data regularly at a fixed interval to the cloud. It is also possible to configure thresholds that when crossed, the device will send a message directly to the cloud. The device also has the option to trigger one of the two output pins once a threshold is crossed.

## 4 Connectors and Interfaces

The Senseceiver 4-20mA has the following connectors:

- Power connector: Terminal block 9 → 48V DC
- Sensors Connector: For connecting up to 4 x 4-20mA sensors and two output pins
- SMA Connector: For connecting a cellular antenna
- USB connector: micro B USB interface for configuring the device and updating the FW

## 5 IO Connector Pinout

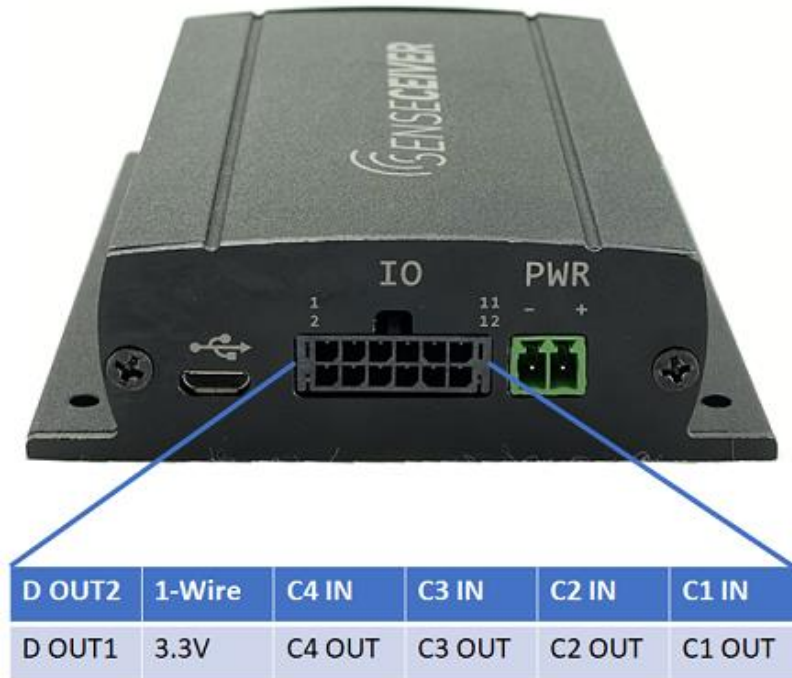


Figure 1

| Pin number | Pin function | Pin description                                            |
|------------|--------------|------------------------------------------------------------|
| 1          | D OUT2       | Digital Output 2                                           |
| 2          | D OUT1       | Digital Output 1                                           |
| 3          | 1-Wire       | Connect to the 1-Wire Sensor data pin                      |
| 4          | 3.3V         | 3.3V to power 1-Wire sensors. Protected by 200mA fuse      |
| 5          | C4 IN        | Connect the output of the fourth 4-20mA Sensor to this pin |
| 6          | C4 OUT       | Fourth sensor current return path                          |
| 7          | C3 IN        | Connect the output of the third 4-20mA Sensor to this pin  |
| 8          | C3 OUT       | Third sensor current return path                           |
| 9          | C2 IN        | Connect the output of the second 4-20mA Sensor to this pin |
| 10         | C2 OUT       | Second sensor current return path                          |
| 11         | C1 IN        | Connect the output of the first 4-20mA Sensor to this pin  |
| 12         | C1 OUT       | First sensor current return path                           |

## 6 Connecting a 4-20mA Sensor

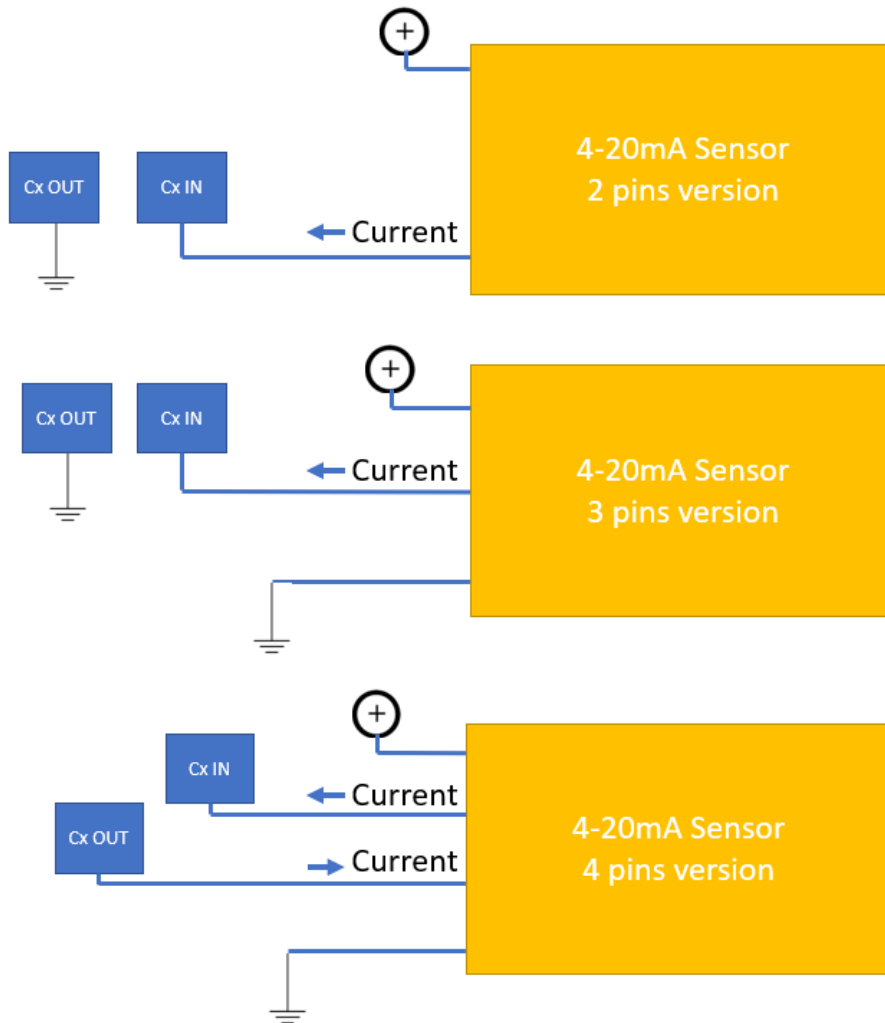


Figure 2

The following figure shows how to connect a 4-20mA sensor to one of the sensors inputs on the Senseceiver 4-20. The current being measures must flow in one of the Cx IN pins and come out from its corresponding Cx OUT pin.

## 7 Connection to Cloud

The Senseceiver can connect to 3 different cloud platforms. You need to decide which cloud to use and make some preparations on the cloud's side before putting the device into operation.

### 7.1 Preparing the AWS IoT Cloud

- Go to this link: <https://eu-central-1.console.aws.amazon.com/iot/home?region=eu-central-1>

- You can select your area for better internet experience. Here we use the EU-Central region.
- You will be asked to login. Login with your username and password.
- After login, you will see the IoT console screen.
- Create a Thing:
  - On the left menu, go to Manage -> Things. Things management UI will appear.
  - On the right-top, there is a Create button, click to create a thing.
  - Select the "Create a single thing" to create.
  - Then Input your Thing name, and then click Next.
  - Now you have a Thing, the Platform may ask you to create or add the certificate for your Thing.
  - Select "one-click certificate creation". After a moment, the Platform will generate the certificate and the key pair for your Thing, save the Thing's certificate and Private Key to your local computer. This is important because you need them to configure our device later.
  - Activate this certificate.
  - Click Done.
- Create a certificate:
  - If you did not create the certificate during your Thing creation, you can create the certificate now follow these steps:
  - On the left menu, go to Secure -> Certificates. The Certificates management UI will appear. Now you can click the "Create" button on the right-top to create the Certificate.
  - Select "one-click certificate creation". After a moment, the Platform will generate the certificate and the key pair for your Thing, save the Thing's certificate and Private Key to your local computer. This is important because you need them to configure our device later.
  - Activate this certificate.
  - Attach Policy, you can click on Attach Policy button at the right-bottom to attach the policy to this certificate.
- Create a Policy:
  - This Policy will allow our device can connect, publish or subscribe to AWS cloud Platform.
  - On the left menu, go to Secure -> Policies. The Policies Management UI will appear, click to the Create button on the right-top to create a new Policy.
  - Input the Policy name, the actions and effects, you can also use a topic filter if you want.
  - The Policy will look similar this:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iot:Connect",
        "iot:Publish",
        "iot:Subscribe"
      ],
      "Resource": "*"
    }
  ]
}
```

```
} ]
```

- We have Thing, Certificate and Policy, now we need to attach them together.
  - Go to Secure -> Certificates
  - Select our Certificate (which is created, each certificate will have the created time, make sure that you select the right certificate)
  - Click on our Certificate, then the new UI will appear.
  - On the left menu, go to Things. New UI will appear.
  - Go to the dropdown menu "Action" on the top-right, select Attach Thing.
  - Attach the Thing we have created before.
  - Same for the policy, back to the Certificate Management UI screen, on the left menu select Policies. The Policies management UI will appear.
  - Go to the dropdown menu "Action" on the top-right, select Attach Policies.
  - Attach the Policy that we have created before.

Now we have a Thing with its certificate and policy on the AWS IoT Cloud.

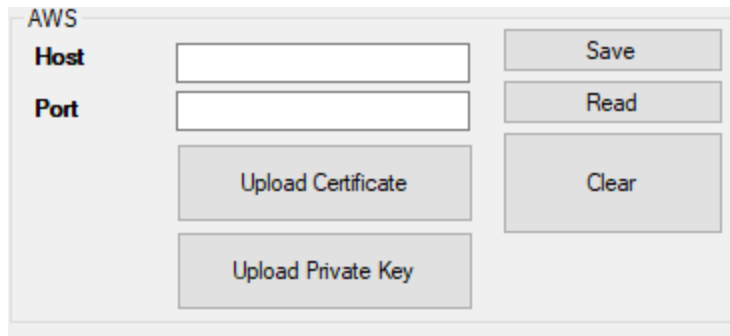
The device must be configured with corresponding host and port. The host could be found under Things/<your device name>/Interact/HTTPS. The used port is 8883.

For example:

Host: <unique endpoint ID>-ats.iot.eu-central-1.amazonaws.com

Port: 8883

You will also need to upload the Certificate and the Private key to the device. You can enter the host and port in this section of the configuration tool and click save. Then you can upload the Certification and the Private Key by clicking on the corresponding button and selecting the corresponding file you have generated in the previous steps.



The screenshot shows a configuration window titled 'AWS'. It contains two input fields: 'Host' and 'Port'. To the right of these fields are three buttons: 'Save', 'Read', and 'Clear'. Below the 'Host' field are two buttons: 'Upload Certificate' and 'Upload Private Key'.

Figure 3

To get data, we have to subscribe to a defined topic. Go to Test tab and fill in the subscribe topic under "Subscribe to a topic" tab with format: "devices/<DID>/messages/events/", with <DID> is the device IMEI.

For example: the subscribed topic for a device with DID "355001096470595" is "devices/355001096470595/messages/events/" (without quotation marks)

After the topic is subscribed, it should be shown under "Publish to a topic" as below:

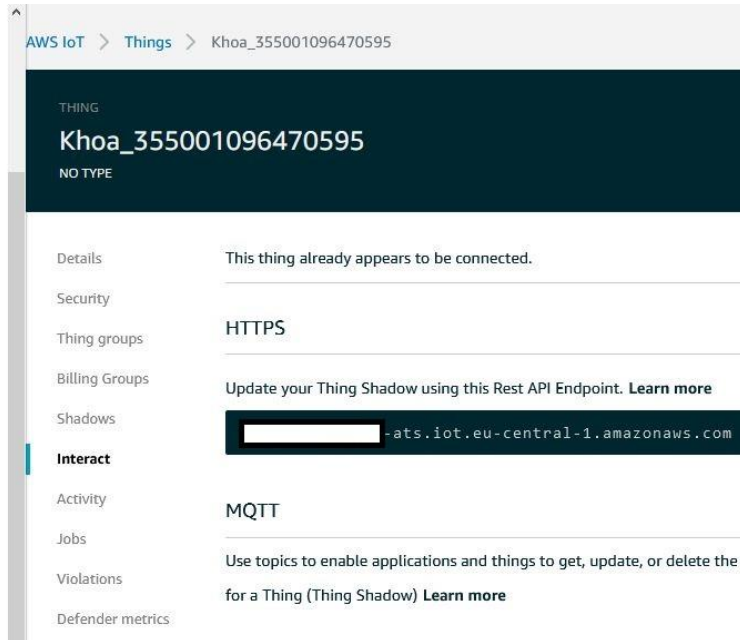


Figure 4

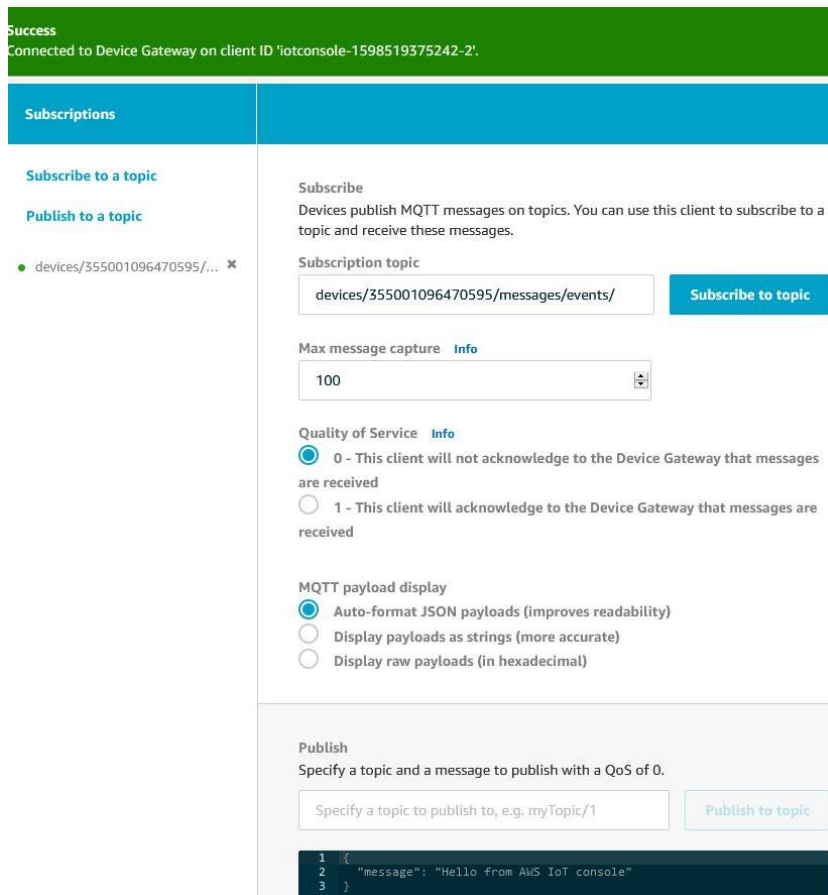


Figure 5

## 7.2 Preparing the Cumulocity Cloud

- You will need to setup an account with Cumulocity. Go to the link below and create your own business account.

<https://www.softwareag.cloud/site/product/cumulocity-iot.html>

- Take a note the device's IMEI, which is printed on the label on the back of the device. This will be the device's ID used in the next steps.
- On the Cumulocity Platform, go to "**Management**" button and then choose **Devices** → **Registration** from the left menu.

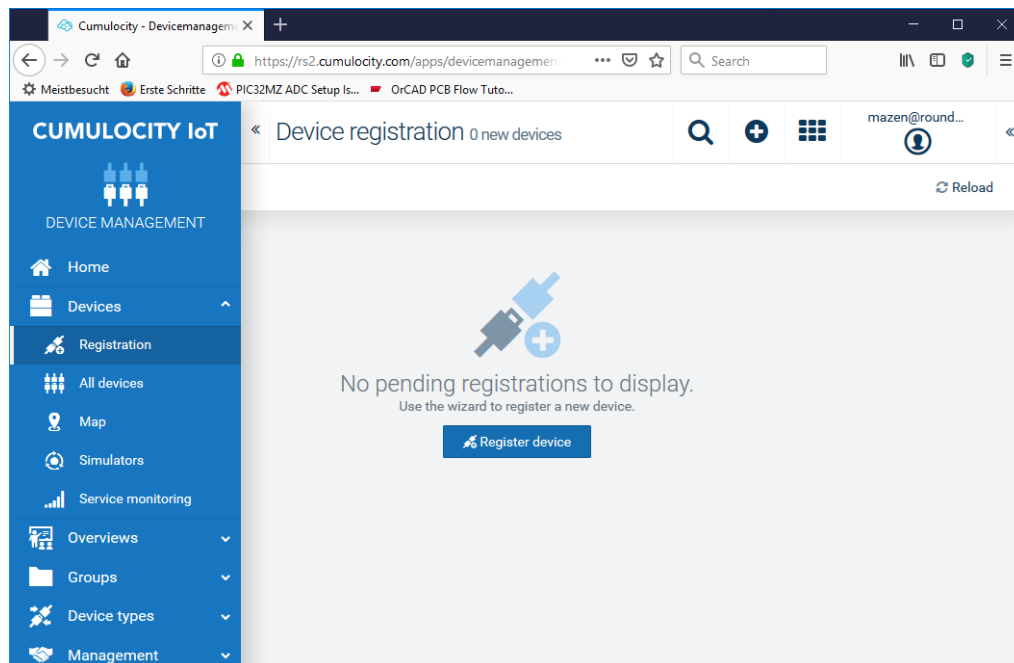


Figure 6

- The screen in Figure (3) should appear. Click on **Register Device** → **General Device** Registration and then enter the IMEI of the device (check back sticker of the device) in the field **Device ID**. Once done, click on **Next** and then **Complete** buttons consecutively. This will put the device in your account under the waiting state till the actual device connects and detects it.

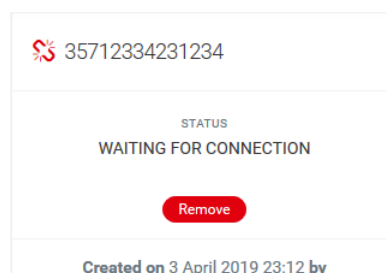


Figure 7

### 7.3 Preparing the Microsoft Azure IoT Hub Cloud:

- Download and Install the Visual Studio Code IDE.
- Run the Visual Studio Code, and Install the Azure IoT Hub extension (as Figure (7)).
- Login your account.

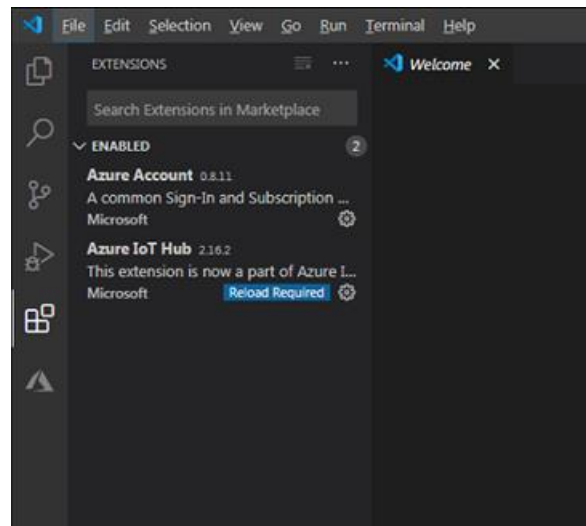


Figure 8

- In explorer tab, Right click on Azure IoT Hub to create a device.

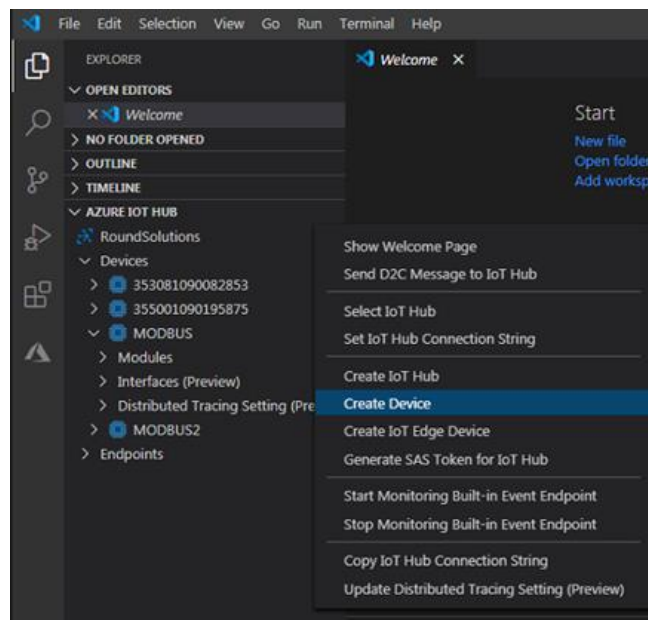


Figure 9

- Device name will be the Device ID. The device should the IMEI printed on the label on the back of the device.

- Generate the device SAS token as Figure (7), you have to enter the expiration time for it.

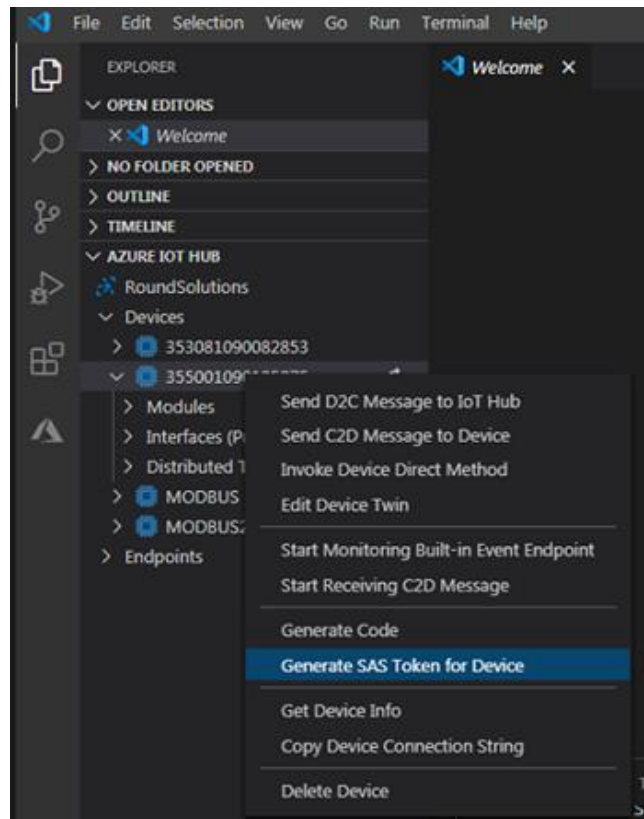


Figure 10

- The generated SAS token will look like this:  
[SASToken] SAS token for [123456789012345] is generated and copied to clipboard:  
SharedAccessSignature sr=xxxxxxxxxxxx.azure-devices.net%2Fdevices%2F123456789012345&sig=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx%3D&se=361591695229
- Then the host, port and credentials for AZURE IoT Hub are:  
Host: "IOTHUBNAME.azure-devices.net"  
Port: 8883  
Username: "IOTHUBNAME.azure-devices.net/DEVICEID/?api-version=2018-06-30"  
SAS Token: "SharedAccessSignature sr=xxxxxxxxxxxx.azure-devices.net%2Fdevices%2F12345678901234&sig=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx%3D&se=361591695229"

Enter the parameters you have obtained so far in the configuration tool in the Azure section shown in the following picture

| Azure     |                                                                                                                                                                                           |
|-----------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Host      | <input type="text"/>                                                                                                                                                                      |
| Port      | <input type="text"/>                                                                                                                                                                      |
| Username  | <input type="text"/>                                                                                                                                                                      |
| SAS Token | <input type="text"/>                                                                                                                                                                      |
|           | <input type="button" value="Save Host/Port"/><br><input type="button" value="Read Host/Port"/><br><input type="button" value="Save Credentials"/><br><input type="button" value="Clear"/> |

Figure 11

## 8 Sensor Threshold Comparison and Output Trigger:

You can configure the device to do edge computing and take actions based on the sensor's value. Using the configuration tool, you can create thresholds and actions. Once a threshold condition becomes valid, data is sent to the cloud and an action is taken on the device accordingly. The following are parameter for setting a threshold condition.

**Sensor ID,threshold,direction,output\_trigger,output\_value,reset**

Where:

- Sensor ID: Sensor ID is from 1 to 4
- Threshold value: The Float number, this value will be compared to the sensor real-time value to determine the output.
- Direction:
  - o **Sensor value less than threshold value**
  - o **Sensor value greater than threshold value**
- Output Trigger: has the following option
  - o No output triggering when the comparison matches the condition.
  - o Trigger Output 1 when the comparison matches the condition.
  - o Trigger Output 2 when the comparison matches the condition.
- Output Value: can be either high or low
  - o targeted output will be driven low if it is triggered
  - o targeted output will be driven high if it is triggered
- Reset:
  - o The output will be reset after the threshold condition is reset.
  - o The output will be remained after the threshold crossing.

Example:

If the threshold data is the following:  
 Sensor= 1, Threshold= 4.5, Direction= greater than, Output trigger = Output 1, Output value = Low, and Reset = yes  
 Then: This threshold is used for Sensor 1.  
 If the Sensor 1 Value is greater than 4.5, then we trigger the output 1 will be released.  
 If the Sensor 1 Value is less than 4.5, then we need to reset the Output 1 to 1 (means the Output 1 will be triggered).

## 9 Sensor Reading:

- The sensor values will be read every second.
- We have totally 4 Sensor Inputs with the sensor ID are 1, 2, 3, 4.
- Each second, the device will read sensor's value, and do the comparison with their thresholds and trigger the outputs.
- We have totally 2 outputs.
- Because the sensor reading interval is 1 second, but the tx\_interval (the interval to send data to cloud via MQTT) is set by user (minimum is 5 seconds). So that, during tx\_interval, we may have maximum (tx\_interval (in second)/1 second) samples of data. We decide to send to cloud the first sample in this data set, and ignore the rest.

## 10 Configuration tool

**Senseceiver 4-20 v1.1**

**File**

Device Connection  
COM Port: COM3  
Buttons: Refresh COM Ports, Close COM Port, Open COM Port, Debug, FW Version, Help

**Basic Configuration**

APN: iot.1nce.net  
Username:   
Password:   
TX Interval: 20  
Cloud Platform: AWS  
Buttons: Save, Read, Clear

**Sensors Thresholds**

| Sensor ID | Threshold | Direction | O/P Trigger | Output Value | Reset |
|-----------|-----------|-----------|-------------|--------------|-------|
| 1         | 0         | Less than | No Trigger  | Low          | No    |

Buttons: Save, Delete Thresholds, Read All

**Sensor Conversion**

Sensor1 = (1 x Current1) + 0  
Sensor2 = (1 x Current2) + 0  
Sensor3 = (1 x Current3) + 0  
Sensor4 = (1 x Current4) + 0  
Buttons: Save, Read, Reset

**FTP**

Host:   
Port:   
Username:   
Password:   
Buttons: Save, Clear

**Azure**

Host:   
Port:   
Username:   
SAS Token:   
Buttons: Save Host/Port, Read Host/Port, Save Credentials, Clear

Developed By: Round Solutions GmbH & CO. KG

Figure 12

The configuration tool should be used to configure the device for proper operation. The following describes the different sections:

- **Basic Configuration:** here you need to set the APN, username and password of the operator. You can also define the data transmission interval in seconds and the cloud to which the data will be sent to.
- **Sensor Conversion:** here you can define the parameters to the formulas used to convert the current reading to actual sensor measurement.
- **Sensor Threshold:** here you can define multiple thresholds for the sensors. Refer to section 8 for more details
- **FTP:** here you can define the FTP server address from where new FW files can be downloaded on the device

- **AWS:** here you can define the AWS host and port, upload the certificate and the private key. Add the parameters to this section only if you are using AWS Cloud.
- **Azure:** here you can define the Azure host and port, and set the credentials (username and SAS Token) of the device. Add the parameters to this section only if you are using Azure IoT Hub.

## 11 LEDs behavior:

- There are 2 LEDs on the device.
- LED 1 behavior:
  - o LED 1 will be in ON state when the device is powered up.
  - o When the device boots up, if the Cellular module is initialized successful, the LED 1 will remain ON state. If the Cellular module is initialized un-successful, LED 1 will go to OFF state.
  - o When the device is fully booted up, It will try to re-initialize the Cellular module if It was un-successful. If:
    - LED 1 is on: Cellular module is initialized successful and the MQTT connection is good.
    - LED 1 is off: The Cellular module is initialized un-successful or the MQTT connection is dropped.
    - LED 1 is blinking: The Cellular module is being initialized or the MQTT connection is being established.
- LED 2 behavior:
  - o LED 2 will be in OFF state when the device is powered up.
  - o When the device is fully booted up, the Sensor reading actions will happen. If:
    - LED 2 is ON: Sensor Reading is performed successful
    - LED 2 is OFF: Sensor Reading is performed un-successful
    - LED 2 is blinking: Sensor Data is being sending to Cloud.

## 12 MQTT Message Structure

- AWS IoT and AZURE IoT Hub Clouds:
  - o The MQTT message will be in json string:
 

```
{
    "Devicename": "Senceiver420",
    "Manufacture": "RoundSolutions.com",
    "SWVersion": "1.0.0",
    "DeviceID": "355001090195875",
    "Temperature": 0,
    "SensorValue1": 0,
    "SensorValue2": 0,
    "SensorValue3": 0,
    "SensorValue4": 0
  }
```
- For the Cumulocity IoT Cloud, we are following the message templates from the Cumulocity Platform. You will automatically receive the data on your Device -> Measurement tab.