AN3083

Fast Prototyping of BLE Sensors for AWS Cloud Using AVR-IoT Development Boards

Introduction

Author: Alin Stoicescu, Microchip Technology Inc.

This application note describes how to connect a BLE (Bluetooth® Low Energy) sensor to the cloud and includes a simple example application. The project is designed for the AVR-IoT WG Development Board which includes an ATmega4808 microcontroller and a temperature and light sensor. The application is intended to be an off-the-shelf system, where the microcontroller can connect to the cloud using an RN4871 Click board™. The Click board is attached on the mikroBUS™ of the AVR-IoT WG Development Board and incorporates a Microchip RN4871 BLE module.

When the application starts, the microcontroller creates two characteristics in the BLE module for the sensors. The microcontroller reads the data from sensors every five seconds and updates the characteristics with the respective values. A gateway is required to transmit the information to the cloud. The gateway establishes a connection with the BLE module of the end node and connects from a local Internet network through Ethernet or Wi-Fi® to the cloud services. Every five seconds the gateway reads the characteristics and publishes the data to the cloud.

A Raspberry Pi (RPi) 3 Model B+ board is used as a gateway since it incorporates BLE, Wi-Fi and Ethernet modules and it is easy to use and provides good documentation.

Amazon Web Services (AWS) is used for cloud services. It provides a gateway core software called AWS IoT Greengrass which allows the users to interact with the cloud. The user can publish and subscribe to topics just by utilizing Application Programming Interfaces (API) in Lambda.

The source code for peripherals used in the microcontroller are generated using Atmel START. The code generator creates the structure of the project, the files, and the source code. Atmel START provides support for every peripheral and development extension boards such as the RN4871 Click board.

The following GitHub repository contains the project source code for the AVR-IoT WG Development Board together with the Lambda used by the gateway to read characteristics and publish the data to the cloud.

Table of Contents

1. Overview

This application note explains how to obtain the hardware and software configurations required by the BLE Internet of Things (IoT) project. The functionality of the example project and the AWS configurations are also detailed.

In the example project, the presented device reads data from the light and temperature sensors. The light sensor provides the light intensity information as an analog signal, while the temperature sensor sends data through an I^2C interface. In order to highlight the transition and evolution of the data, a five second reading interval was used for transmitting the data to the cloud. The device exchanges information with the RN4871 BLE device through the USART interface and starts by creating services and characteristics for sensors and updates them after every reading.

The Raspberry Pi board connects to the Bluetooth device, reads the characteristics and, using the Greengrass core, publishes the data to the cloud. Greengrass is a gateway software provided by AWS capable of using the hardware components of the Raspberry Pi board. Thus, the gateway core can interact with nearby BLE node devices and can provide secure communication to the cloud.

Prerequisites:

- Integrated Development Environment (IDE):
  1. Atmel Studio v7 (used in this project) with the latest device packages installed
  2. MPLAB® X v5.15 with XC8 v2.05 compiler
- Microchip AVR-IoT WG Development Board
- Bluetooth Low Energy RN4871 Click board from MIKROE
- Raspberry Pi 3 Model B+
- Amazon Web Services (AWS) Account

2. Hardware Description

2.1 AVR-IoT WG Development Board

The AVR-IoT WG Development Board is intended to be used in the Internet of Things environments. The board incorporates a multitude of IoT oriented devices, such as an ATmega4808 microcontroller, an ATWINC1510 Wi-Fi module for communications, an ATECC608A CryptoAuthentication™ device for security, a TEMT6000 light sensor and an MCP9808 temperature sensor for data acquisitions.

The board integrates four user LEDs and two user buttons for a better user experience and an easier interaction with the application, and a Lithium Polymer (LiPo) charger so the board can be used on the field. The board also features a mikroBUS slot, offering diversity and allowing the user to enhance the features of their system. The user can add or stack several Click boards from sensors to actuators or even new communication modules, as presented in this application note. In this project, a BLE click module was connected to the AVR-IoT WG Development Board.

The board can be powered from a USB or an external LiPo battery connected to the J101 battery plug. The ATmega4808 microcontroller and the RN4871 Click board are powered at 3.3V. More details about the schematic can be found on the following link: http://ww1.microchip.com/downloads/en/DeviceDoc/AVR-IoT_WG_Schematics.pdf.

Microchip's ATWINC1510 is an IEEE® 802.11 b/g/n IoT network controller. It is the ideal add-on to existing MCU solutions bringing Wi-Fi and Network capabilities. The ATWINC1510 provides internal Flash memory as well as multiple peripheral interfaces including UART and SPI.

Microchip's ATECC608A is a secure element with advanced Elliptic Curve Cryptography (ECC) capabilities. It integrates the Elliptic Curve Diffie-Hellman (ECDH) security protocol which is an ultra-secure method to provide key agreement for encryption/decryption, along with the Elliptic Curve Digital Signature Algorithm (ECDSA) sign-verify authentication for the IoT systems. In addition, the ATECC608A offers an integrated Advanced Encryption Standard (AES) hardware accelerator strengthening the hardware based security.

The board was chosen because it encourages IoT development, but since the purpose of the application note is to transmit BLE data to the cloud, the ATWINC1510 module and ATECC608A CryptoAuthentication™ device will not be used in this project.

2.2 MIKROE RN4871 Click Board

The RN4871 Click board is a BLE board which can be inserted in the mikroBUS slot. It is powered from 3.3V and is connected to the host microcontroller by using the UART peripheral, allowing control via ASCII commands. The board also provides support and channels for Ready To Send (RTS) and Clear To Send (CTS) UART signals, which are optional and will not be used in this application.

The RN4871 is a Bluetooth low-energy module. It has a completely integrated software based on the Bluetooth stack version 4.2, which delivers up to 2.5x throughput improvement and more secure connections than Bluetooth 4.1 based products.

The module supports transparent UART which is used for data transfer from UART to a peer BLE device. The default baud rate is 115200 symbols/s, but it can be changed from the command interface. The data exchange over UART is done through two channels, one for transmitting and one for receiving.

The module supports Generic Attribute Profile characteristics (GATT) and it can implement up to five public and four private user-defined GATT services, each service providing up to eight characteristics.

When interfaced with a BLE enabled smartphone or Bluetooth Internet Gateway, the applications can be monitored, controlled and updated from anywhere in the world. Thus, the RN4871 module is perfect for IoT applications.

2.3 Raspberry Pi 3 Model B+ Board

The Raspberry Pi 3 Model B+ Development Board runs various distributions of the Linux™ operating system, granting open source software libraries, drivers and applications to the users. These allow easy interfacing with other devices. The board provides a built-in BLE module for communication with the end nodes and Wi-Fi and Ethernet socket for the cloud connection and communication, favorable for IoT. The RPi 3 B+ has an armv71 processor architecture and will be used in this demo application as a gateway. The user will need a microSD card which is at least 8 GB in size for the operating system of RPi and the gateway software.

3. Application Overview

3.1 Software Requirements

Microchip's MPLAB® X and Atmel Studio are Integrated Development Environments (IDEs) used to develop applications for microcontrollers. The application presented in this document has been developed in Atmel Studio v7, but the project can also be imported in MPLAB X from *File → Import → Atmel Studio Project*. The MPLAB X IDE v5.15 and the XC8 compiler v2.05 are required for the correct functionality of this project.

Atmel Studio v7 IDE gives a seamless and easy-to-use environment to write, build, and debug user applications written in C/C++ or in assembly code. Atmel Studio v7 supports all 8-bit and 32-bit AVR microcontrollers, the new SoC wireless family, SAM microcontrollers, and connects seamlessly to Atmel debuggers and development kits.

Microchip provides a tool for generating code called Atmel START which is available in Atmel Studio and as a web-based software. Both versions of Atmel START require an Internet connection. Atmel START supports code project generation for Atmel Studio v7, IAR Embedded Workbench®, or generic makefile generation.

The user must make sure that Atmel Studio has the latest device packages installed. This can be done through the following steps: the user must click on *Tools → Device Pack Manager*, then press '**Check for Updates**', and in the '**Install**' tab select '**Install all updates**'.

3.2 Amazon Web Service Software Overview

To use Amazon Web Services, the user needs an account. This can be created for free and allows the use of the gateway software and AWS for up to one year for multiple devices. The cloud account and gateway (which is called Greengrass) configurations are not the focus of this document and will not be explained in detail. AWS already provides a well-documented and written developer guide which can be found at Getting Started with AWS IoT Greengrass. The Greengrass developer guide is divided in chapters, based on the focus of the information presented. All of these chapters are called modules.

Module 1 explains how to download and load the operating system for Raspberry Pi, how to establish a connection between the board and the computer. This module also explains step by step what configurations have to be done so the board can safely run the Greengrass.

Module 2 describes how to download and install the gateway software on Raspberry Pi. It comes with public and private keys and a certificate which are mandatory for cloud connection. This module provides details on how to start the gateway and how to check the process which handles the gateway.

Therefore, these two modules are mandatory for configuring the Raspberry Pi board and getting started with the Greengrass core.

Module 3 is focused on Lambda and data exchange between the cloud and the RPi. The module explains how to add and deploy Lambda on Raspberry Pi. AWS Lambda is a function which runs code without provisioning or managing servers. The Lambda can access the cloud by sending Message Queuing Telemetry Transport (MQTT) messages on specific topics. The sending procedure is called publishing. The client can subscribe to a specific topic and receive the messages.

Module 3 is also required since the purpose of the application presented in this document is to send BLE data to the cloud and this will be implemented by using Lambda. The other modules are not required nor used throughout this application note.

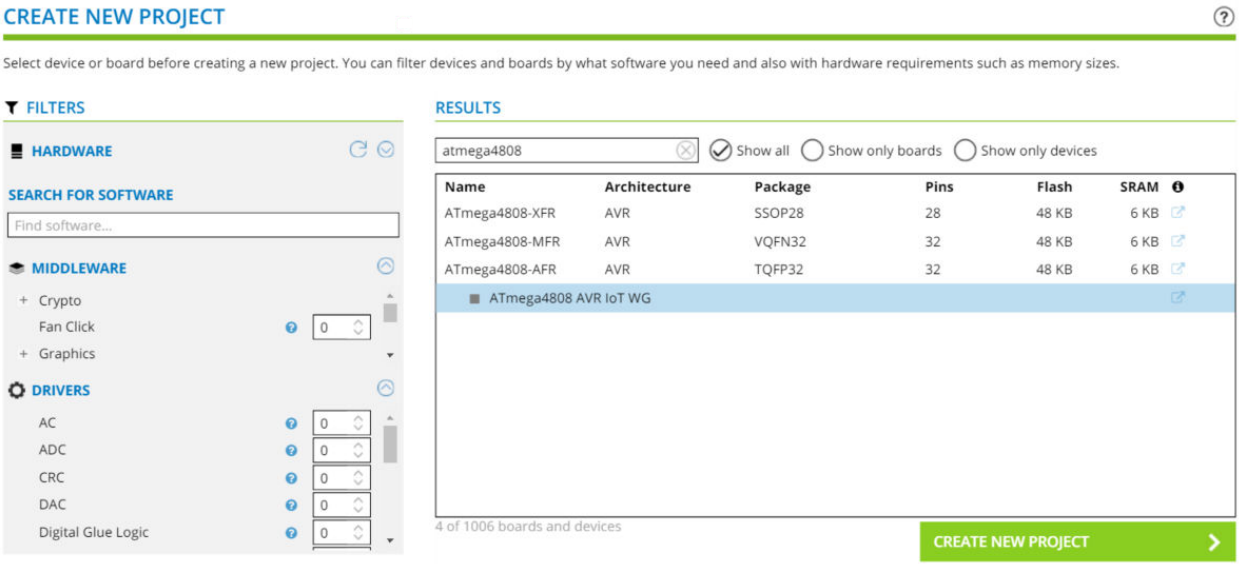
Note: Greengrass core (GGC) v1.7 was used in this project.

3.3 AVR-IoT WG Development Board Software Overview

The project is developed using Atmel Studio and the peripherals required were configured using Atmel START. The user can create a new project by pressing *File → New → Atmel Start Project* within Atmel Studio IDE. The user must filter the devices and select ATmega4808 which is the microcontroller used on the AVR-IoT WG Development Board.

The whole project is provided in the GitHub repository. The purpose of this chapter is to familiarize the users with Atmel START Code Generator and with the structure of the generated code and to explain the usage of every peripheral and its configurations.

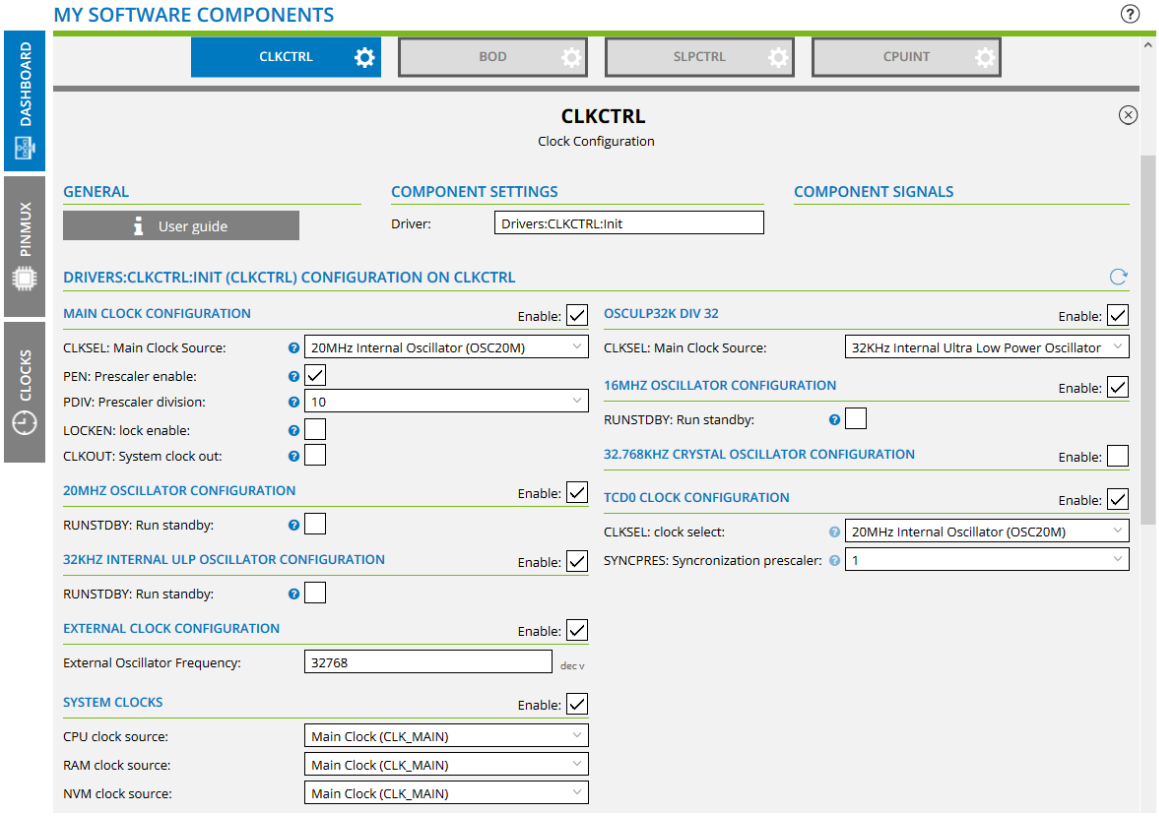
Figure 3-1. AVR® START Project



The peripherals have to be configured as follows:

1. Configure the Main Clock from the CLKCTRL driver as follows:
   - CLKSEL (Main Clock Source): 20 MHz Internal Oscillator
   - PEN (Prescaler Enable): Check Box
   - PDIV (Prescaler Division): 10

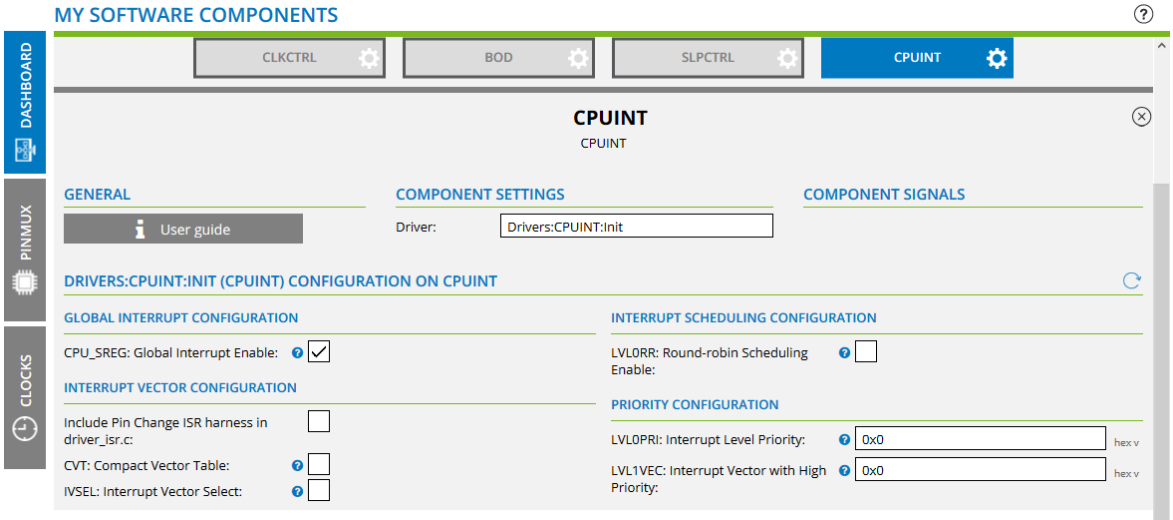
Figure 3-2. Clock Configuration



Note: The resulted Main System Clock has a frequency of 2 MHz.

2. Configure the interrupts from the CPUINT driver:
The interrupts are used for timer and USART data exchange so the communication can take place whenever it is necessary. To enable the interrupt functionality, the global interrupts need to be enabled:
   - CPU_SREG (Global Interrupt Enable): Check Box

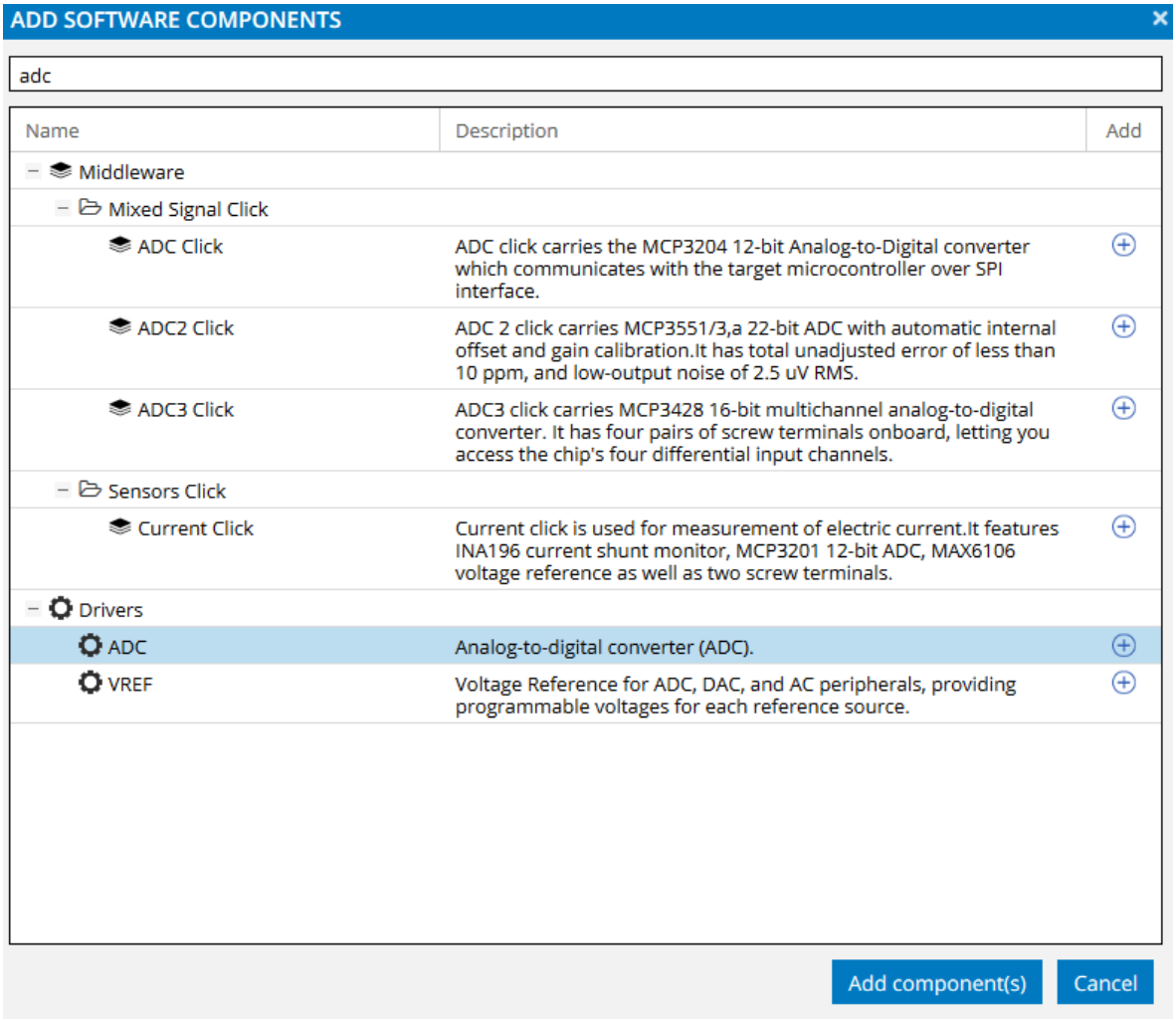
Figure 3-3. Interrupts Configuration



3. Adding and configuring the ADC driver:

This can be done by pressing '**Add software component**' and searching for '**ADC**'.

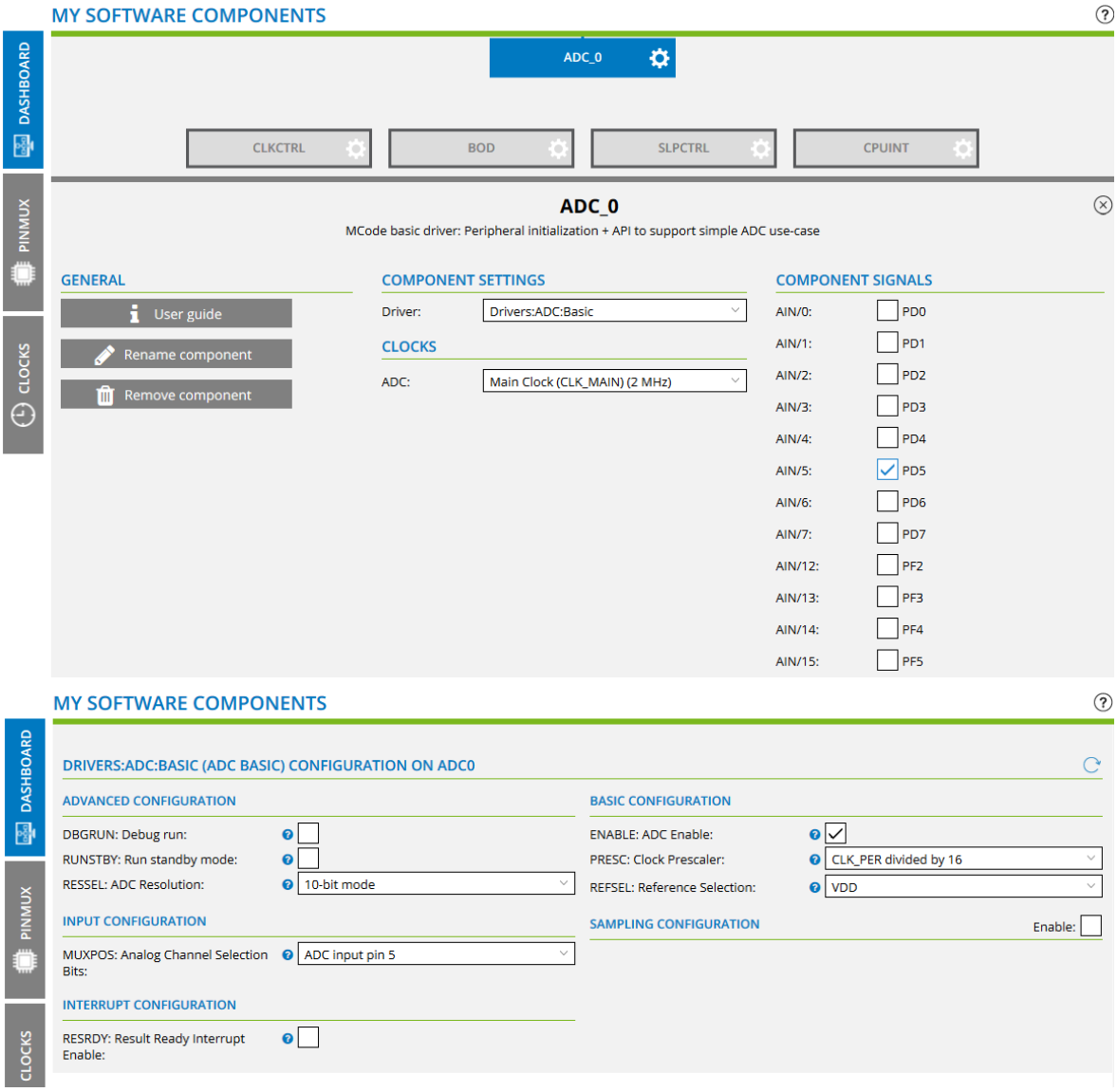
Figure 3-4. Adding ADC Software Components



ADC is used to read the analog output generated by the light sensor and must be configured based on the AVR-IoT WG Development Board current pinout configuration, as follows:

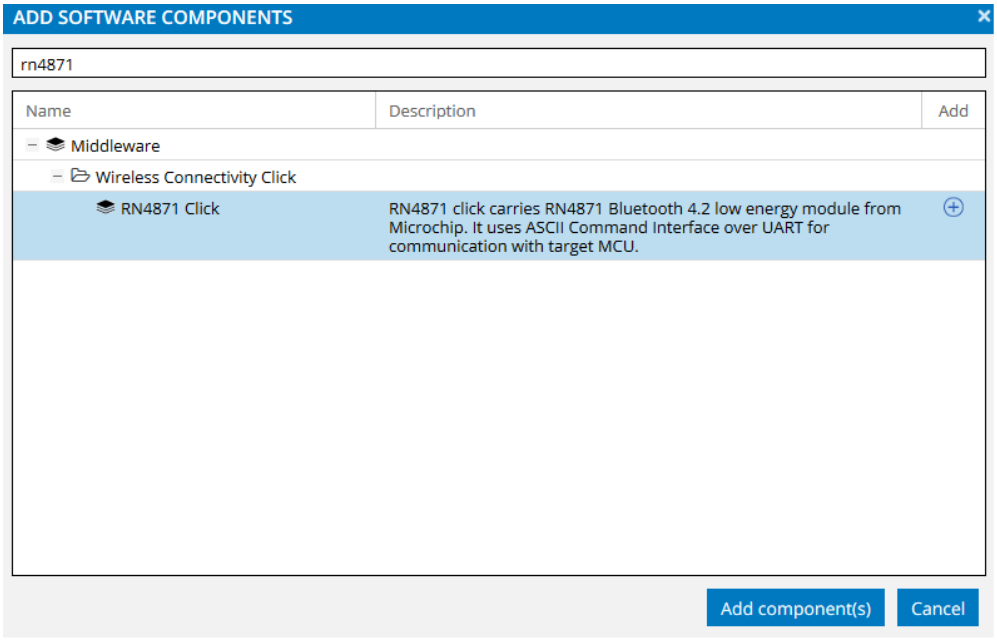
- Driver: Drivers: ADC: Basic
- AIN/5: Check box PD5
- RESSEL (ADC Resolution): 10-Bit mode
- MUXPOS (Analog Channel Selection Bits): ADC input pin 5
- ENABLE (ADC Enable): Check box
- PRESC (Clock Prescaler): CLK_PER divided by 16
- REFSEL (Reference Selection): V_{DD}

Figure 3-5. ADC Configuration



4. Adding and configuring the USART driver for RN4871:
   Microchip provides support for the RN4871 Click board. This can be done by pressing '**Add software component**' and searching for '**RN4871**'.

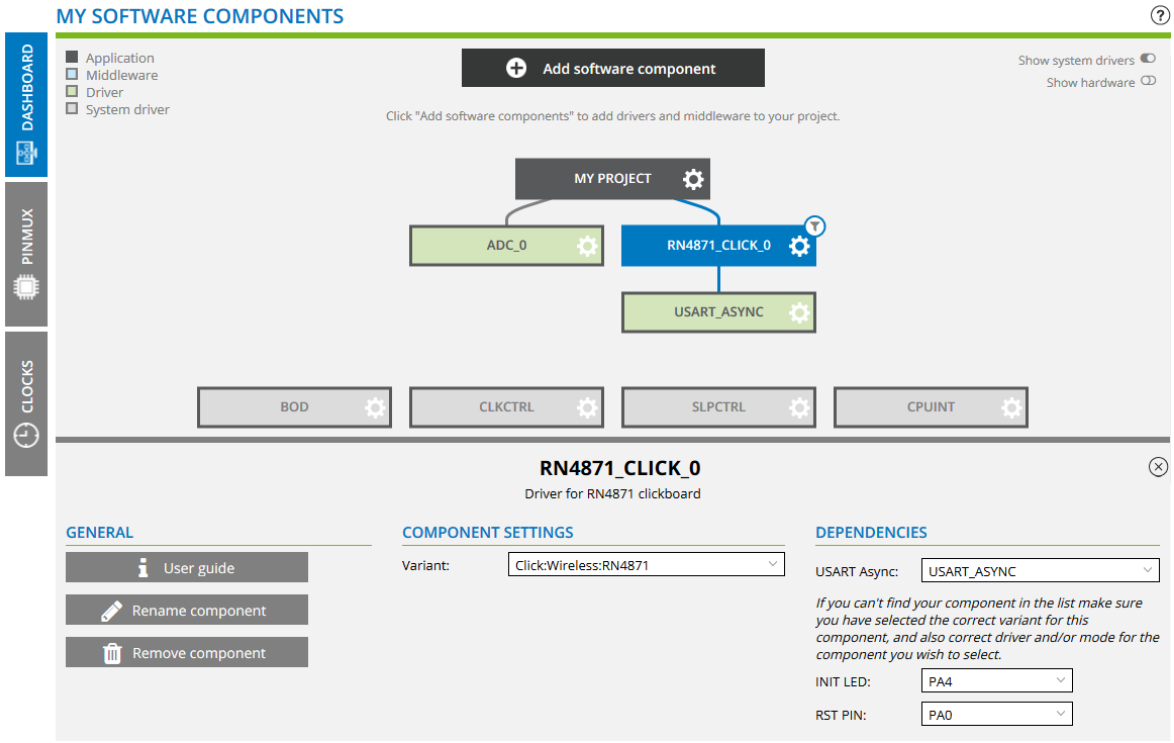
Figure 3-6. Adding RN4871 Software Component



The code generator will provide most of the configurations required by this board. The only changes that must be implemented by the user are related to the location of the pins so these can match the mikroBUS. In order to be able to recognize this instance of USART, it will be renamed to USART_RN4871. The global interrupts have been enabled above and are meant for communication, so the USART must be set to Interrupt Request (IRQ) mode:

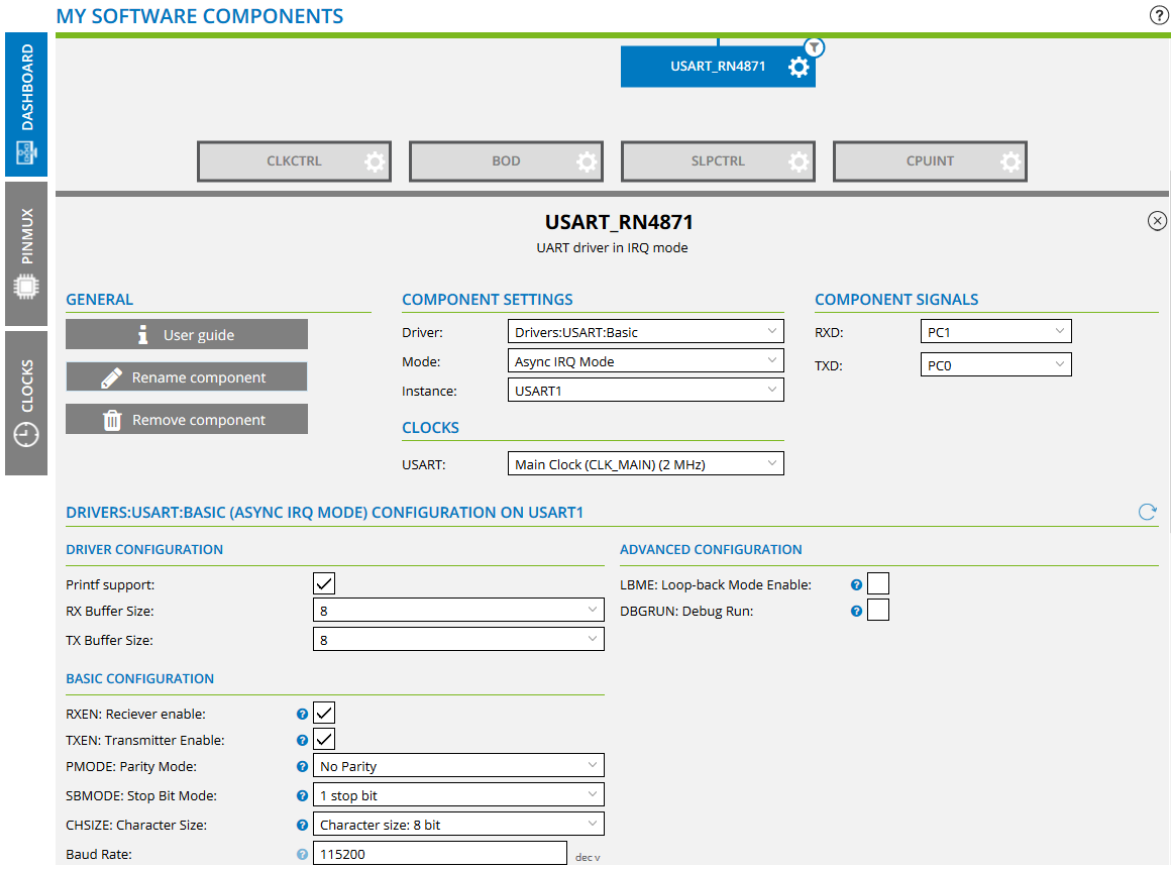
- RST PIN: PA0

Figure 3-7. RN4871 Pins Configuration



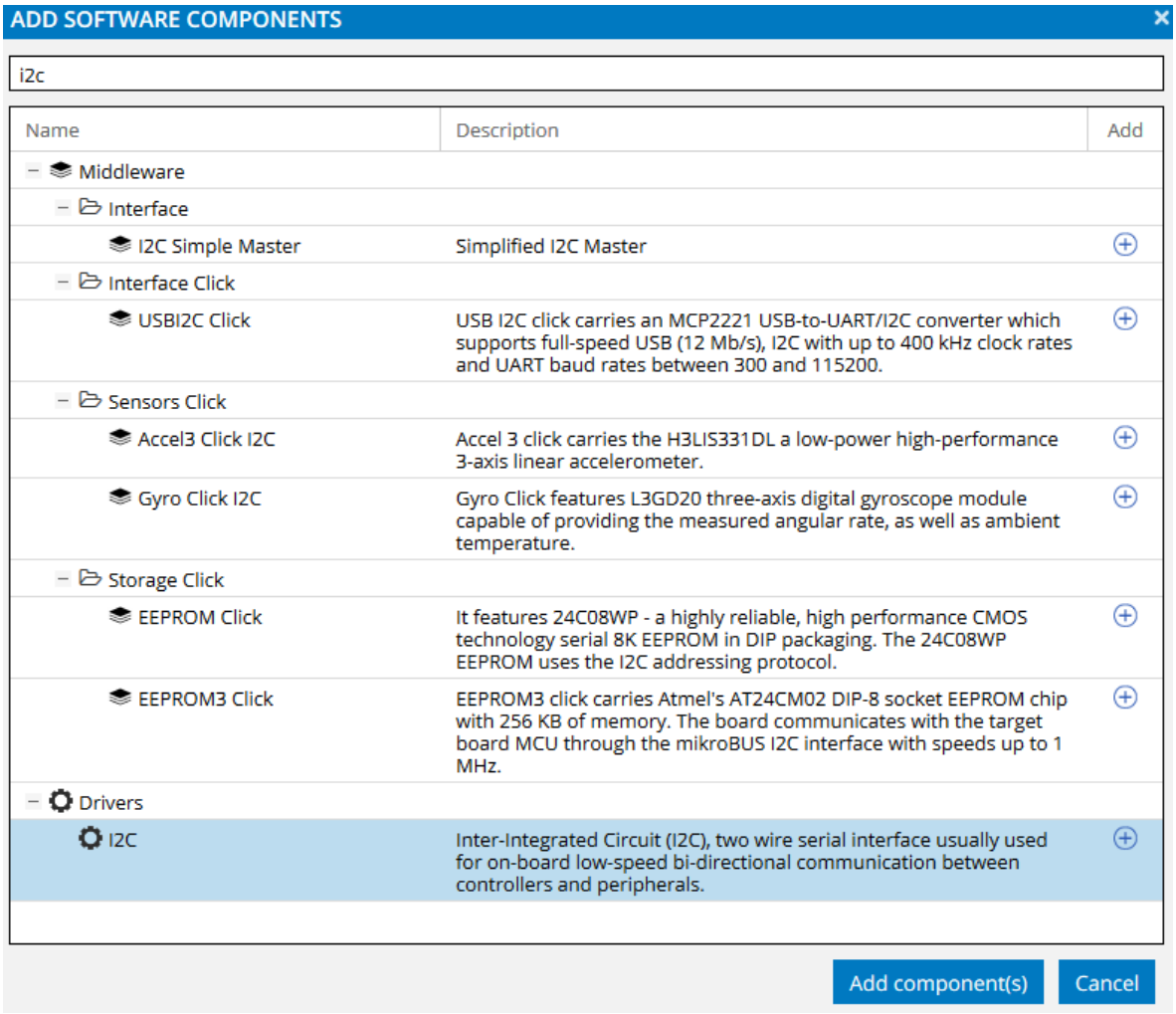
- Rename component: USART_RN4871
- RXD: PC1
- TXD: PC0
- Mode: Async IRQ mode

Figure 3-8. RN4871 USART Configuration



5. Adding and configuring the I²C driver:
This can be done by pressing '**Add software component**' and searching for '**I2C**'.

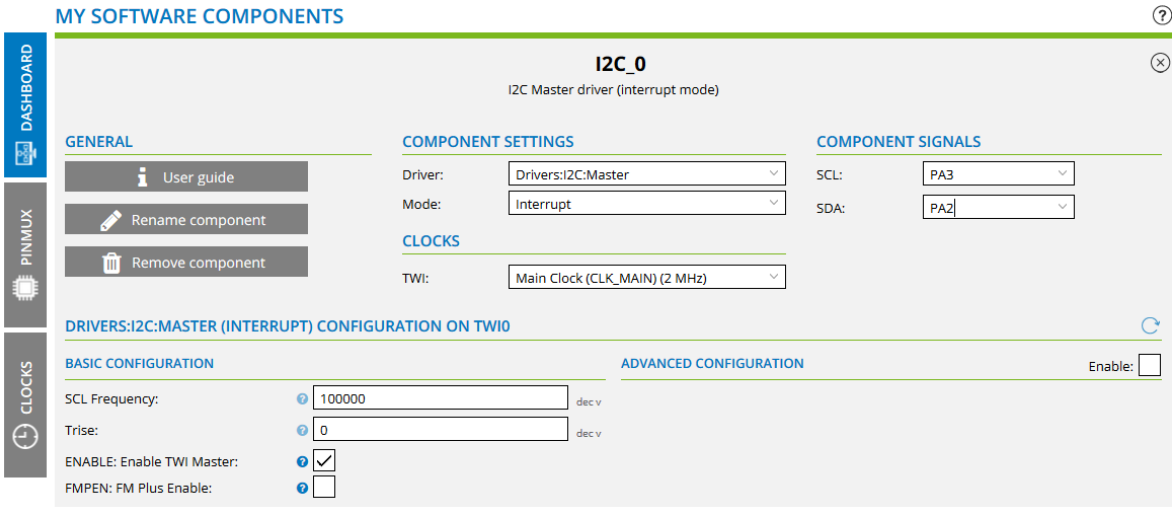
Figure 3-9. Adding I²C Software Component



I²C is used as a master to read the value generated by the temperature sensor and must be configured as follows:

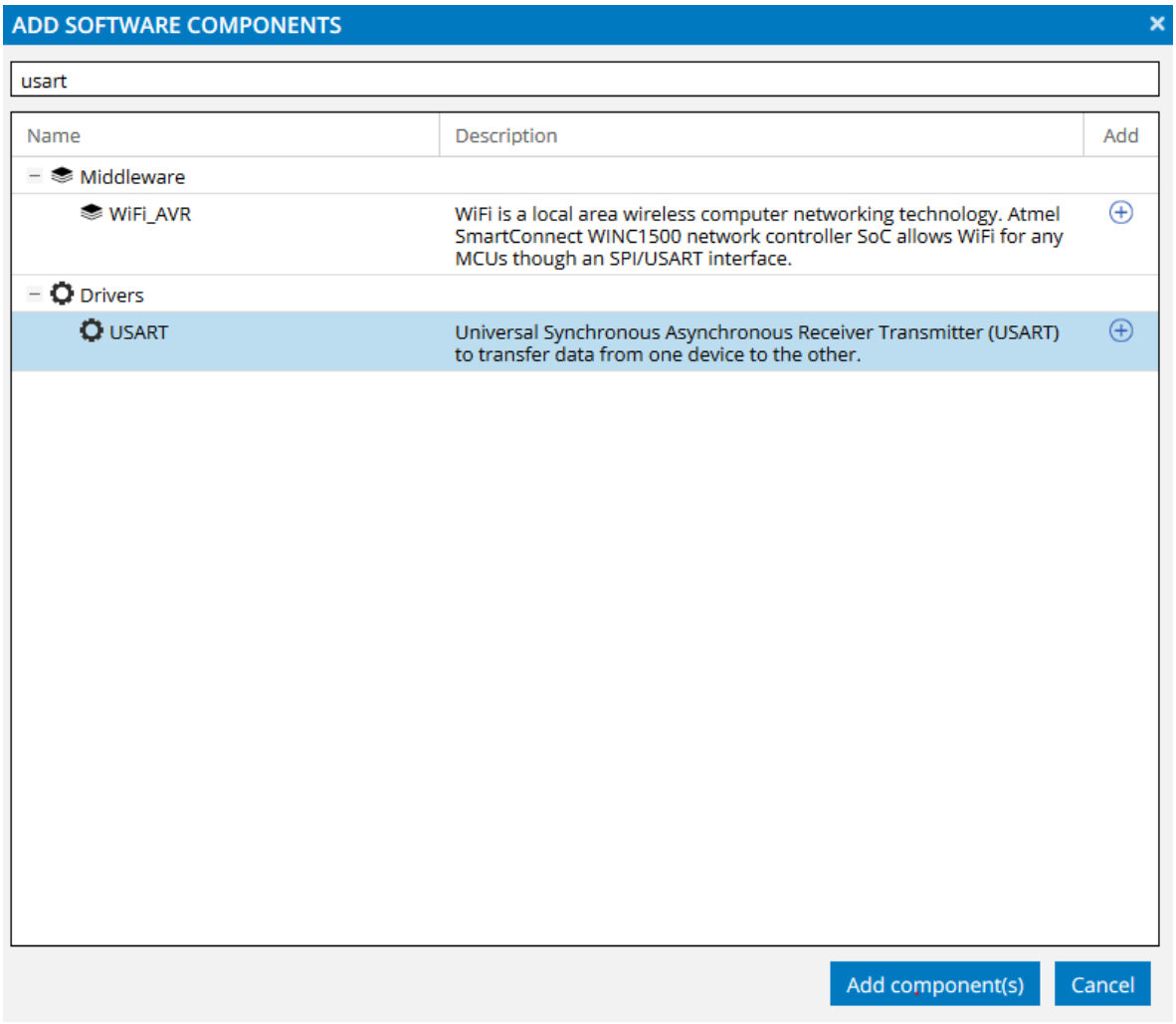
- Driver: Drivers: I2C: Master
- Mode: Interrupt
- SCL: PA3
- SDA: PA2
- SCL Frequency: 100000
- Trise: 0
- ENABLE (Enable TWI Master): Check Box

Figure 3-10. I²C Configuration



6. Adding and configuring the USART driver for PC:
This can be done by pressing '**Add software component**' and searching for '**USART**'.

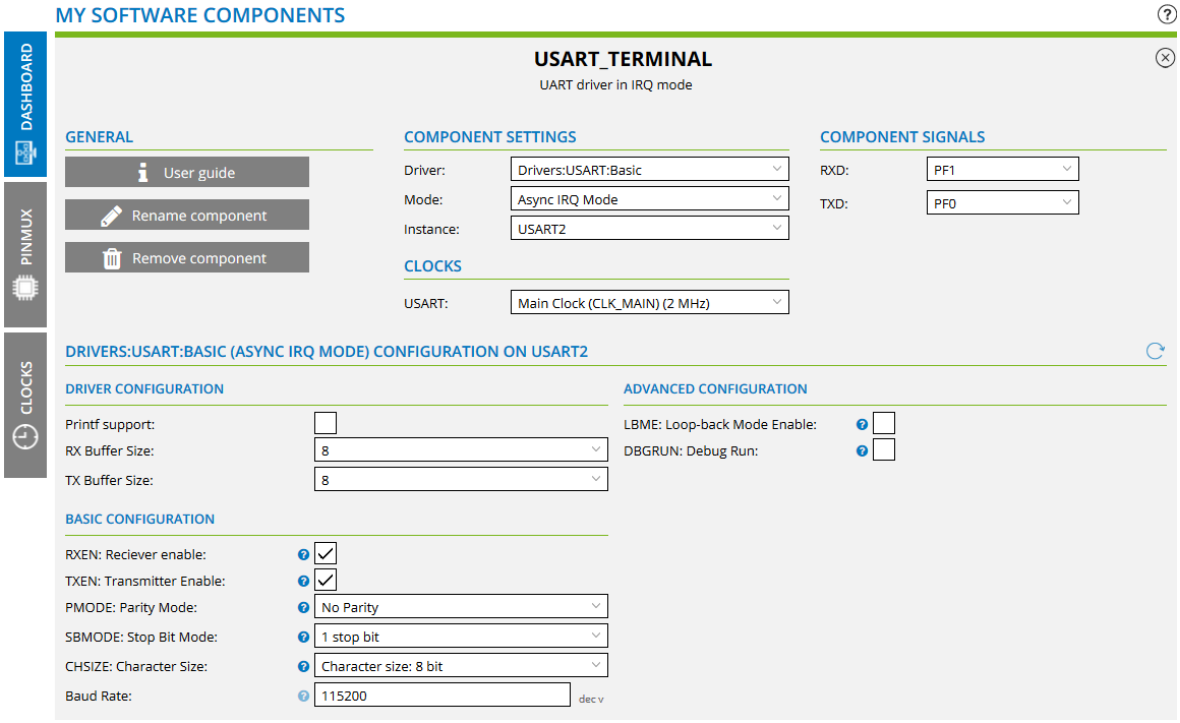
Figure 3-11. Adding USART Software Component



This instance of USART is used to exchange data with a computer. The AVR-IoT WG Development Board contains a debugger which can also act as an USB to TTL converter, meaning that the user can open a serial COM port on the computer and the microcontroller will receive data in its USART peripheral. This communication will mostly be used for RN4871 configurations such as: creating services and characteristics, obtaining the handlers of the characteristics and obtaining the MAC address of the BLE device. This communication can also act as a user interface for logs or commands if the user desires. The baud rate can be changed to any value as long as it is the same value on the PC side in COM port. In order to be able to recognize this instance of USART, it will be renamed to USART_TERMINAL. This instance of USART must be configured as follows:

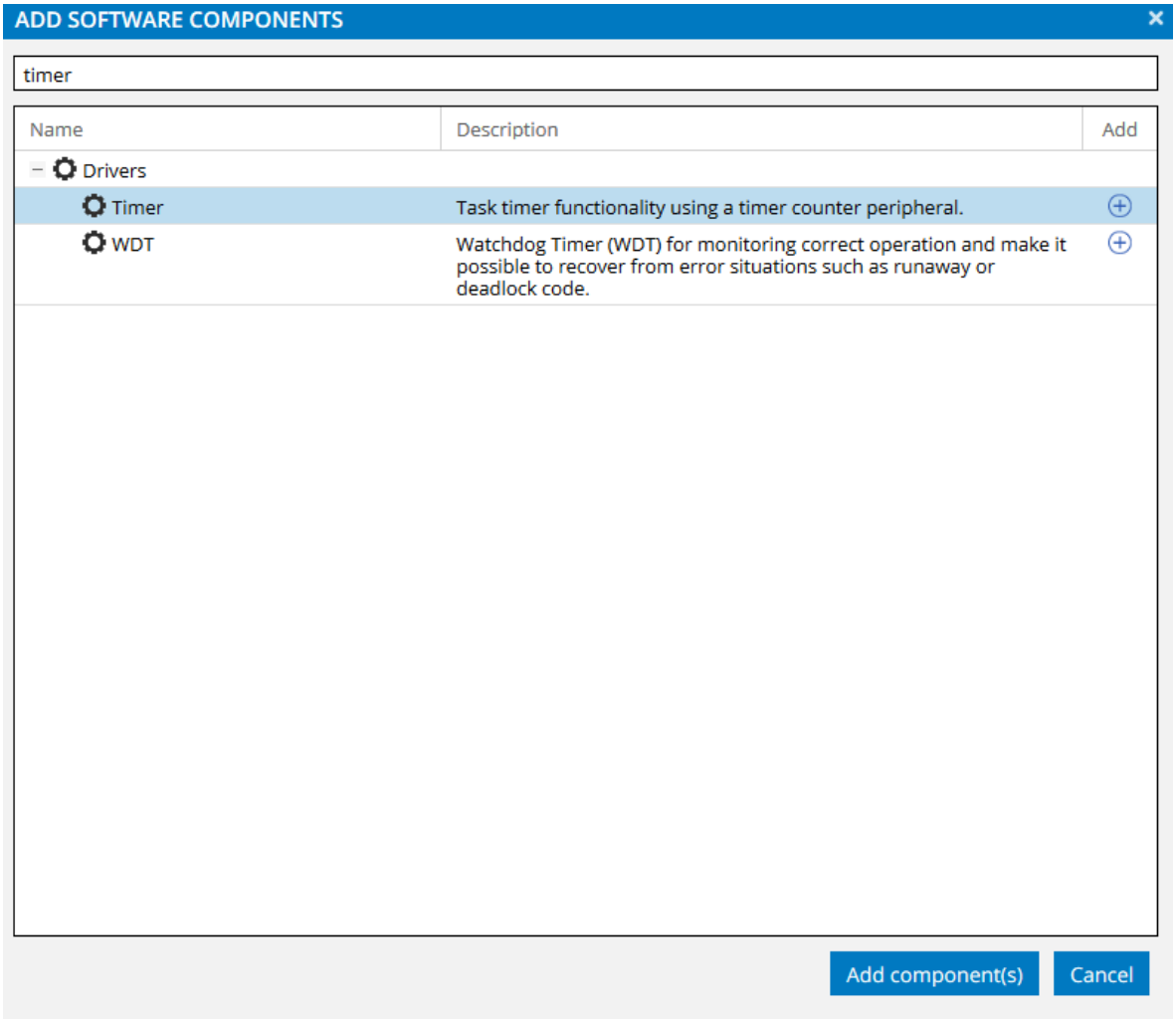
- Rename component: USART_TERMINAL
- RXD: PF1
- TXD: PF0
- Mode: Async IRQ mode
- Baud Rate: 115200

Figure 3-12. PC USART Configuration



7. Adding and configuring a timer driver for sensor reads:
   This can be done by pressing '**Add software components**' and searching for '**Timer**'.

Figure 3-13. Adding Timer Software Component



The timer will be used to introduce a delay between the application tasks. The purpose of the timer will be to count five seconds and when the time has passed to generate an interrupt. When the interrupt arrives, the application will read the sensors and will update the characteristics of the RN4871 BLE module and then wait for the next interrupt.

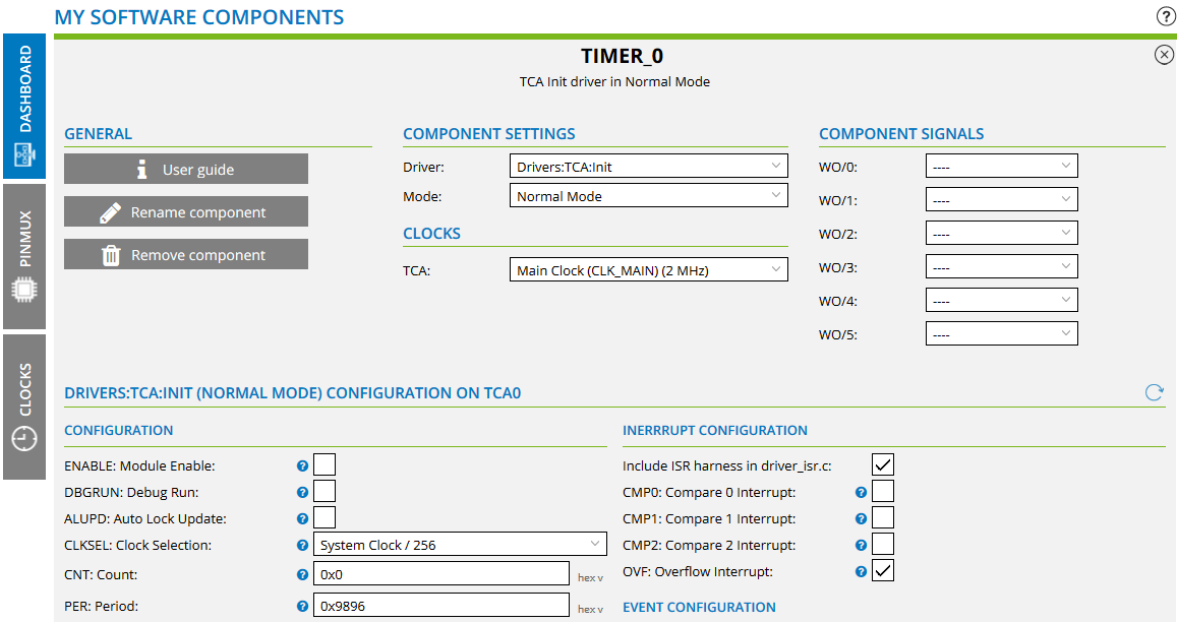
The source of the timer will be the Main System Clock which is configured at 2 MHz. The following equation is used to obtain the values of the PER register.

$$PER = CLKSEL \times 5s = \frac{SYSTEM\_CLOCK}{256} \times 5s = \frac{2MHz}{256} \times 5s \cong 39062 = 0x9896$$

The timer must be configured as follows:

- CLKSEL (Clock Selection): System Clock/256
- PER (Period): 0x9896
- Include ISR harness in driver_isr.c: Check Box
- OVF (Overflow Interrupt): Check Box

Figure 3-14. Timer Configuration



After configuring all these peripherals the project is ready to be generated and the user can press the '**EXPORT PROJECT**' button and then '**DOWNLOAD PACK**' for the web-based Atmel START version or just press '**GENERATE PROJECT**' from the bottom of the page in the Atmel Studio version. The code generator will create the structure and the files of the project and the only requirement left for the user is to provide the main application.

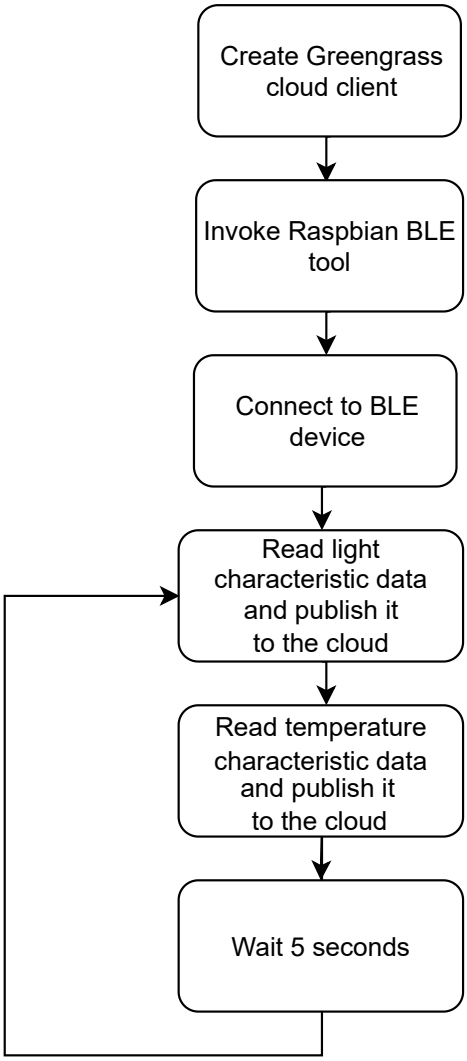
3.4 Lambda Overview

Lambda is one of the ways in which the Greengrass gateway can interact with end devices and with the cloud. A Lambda is configured or edited in the cloud and deployed on the Raspberry Pi board. The function comes with a Software Development Kit (SDK) which provides Application Programming Interface (API) used for cloud connection and topic-specific subscriptions and publications. The messages between the Raspberry Pi and the cloud are sent through the MQTT protocol, but the Greengrass core takes care of the procedures and therefore, these details are transparent for the user.

The Lambda can be written in several programming languages and versions of programming languages. The one used in this application is written in Python™ 2.7 since *Module 3* of the tutorial provided by AWS for Greengrass is based on Python 2.7 and thus it will be easier for the user to understand and get familiarized with the structure and the procedures of the Lambda.

The flow diagram of the Lambda is described below:

Figure 3-15. Lambda Workflow



1. The Lambda will create a Greengrass client responsible for cloud communication.
2. It will invoke a tool responsible for BLE connection and communication, and will enter its virtual user interface. Other BLE oriented tools that are available on Raspbian can be used.
3. It will connect to the BLE end device using its MAC address. The tool does not provide device scanning and the MAC address is different for every device so the user is responsible for providing it. More information on how to obtain the MAC address can be found in 4.2.2 Command State.
4. Once the connection has been established, it will read the bytes available in the temperature and light characteristics using the '`char-read-hnd`' command followed by the handler number, will pack the data and will publish it to the cloud with the topic 'BLE/data'. This step is in a loop and the actions are executed once every five seconds. The topic name and the time between executions are the user's choice and can be changed. The read of the characteristics is done through a handler which is assigned when the characteristic is created. More information on how to obtain the handler can be found in 4.2.2 Command State.

4. Application Demo

4.1 Main Application Configurations

The Atmel START code generator will create the peripherals files and APIs together with the structure of the project and the initialization of the system. The user must add the Reset sequence of the BLE module to make sure everything starts normally, must write functions to read the values of the sensors, and must interconnect the signal lines of the USART peripherals so the BLE module can be controlled from the PC. The user must read the sensors and update the characteristics every once in a while. All these steps have been already written and provided in the GitHub repository project.

4.2 Getting the Board Ready

The user must insert the RN4871 Click board in the mikroBUS of the AVR-IoT WG Development Board and then connect it through a micro-USB to a laptop. The main project, downloaded from the GitHub repository, can be found in the 'AVR-IoT_WG_with_BLE' folder and can be opened by double clicking the '**`AVR_BLE_Project.atsln`**' file. The user can program the board directly using Studio or by dragging and dropping the hex file in the mass storage device associated with the AVR-IoT WG Development Board.

The project is divided in two states: the Application state, in which the application reads the sensors and updates the characteristics values every five seconds, and the Command state, which can be used to interact with the RN4871 board.

4.2.1 Application State

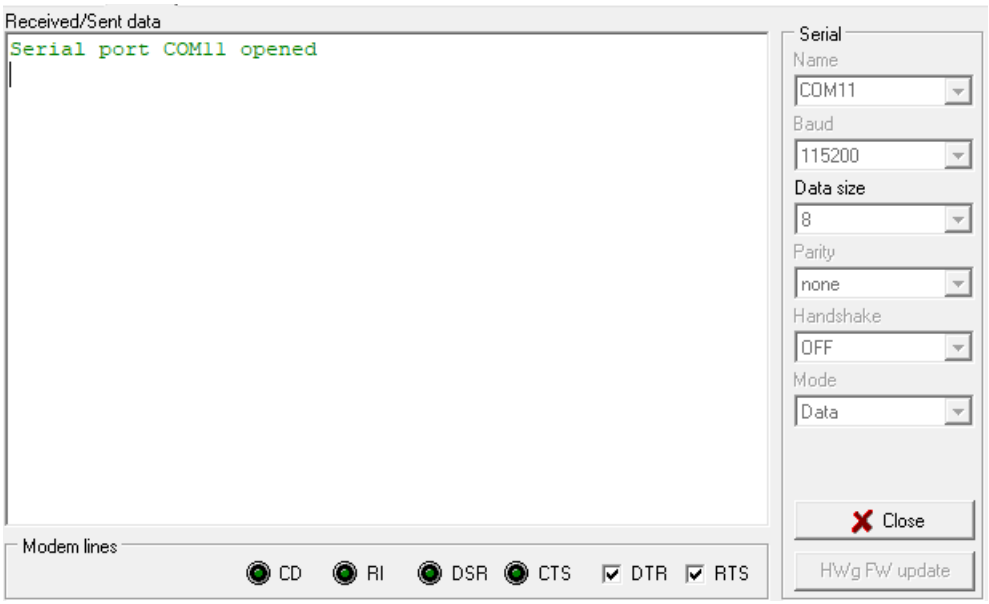
The application starts in Application state and it begins by initializing and checking the configurations of the RN4871 module. It creates services and characteristics for the light and temperature sensors if these do not already exist. Every five seconds, the application reads the sensors and updates the characteristics with the values of the sensors.

4.2.2 Command State

The Command state allows the user to send commands directly and to check the functionality of the RN4871 module. The Command state can be entered at any moment by sending the '/' character in a serial software terminal, running on the PC (such as Hercules or TeraTerm). When using this character, the application replies the new state to the user.

Before opening the COM port, the user must make sure the baud rate is the same as the one selected in Atmel START for the second USART instance (115200 baud rate). In this application note the COM port is opened using the Hercules terminal. The user must make sure the DTR and RTS are activated since the debugger from the AVR-IoT WG Development Board requires flow control.

Figure 4-1. Hercules Terminal



In Command state, the user must use the following commands and:

1. Check for the MAC address of the device. This info will be required later by the Lambda. The MAC address will be on the first line of the reply:
   - Send: 'd'
   - Reply: 'BTA=D88039F37559'
   - Reply: 'Name=RN4870-7559'
   - Reply: 'Connected=no'
   - Reply: 'Authen=2'
   - Reply: 'Features=0000'
   - Reply: 'Services=C0'
2. Make sure the board uses the latest firmware available, which at this moment is v1.30:
   - Send: 'v'
   - Reply: 'RN4871 V1.30 3/18/2018 (c)Microchip Technology Inc'

In Command state, the user can utilize any command described in BLE RN4871 Module User's Guide.

Note: In this state, the application is no longer reading sensors and updating the characteristics, so the user must send again the '/' character to switch back to Application state.

4.3 Creating and Loading Lambda

The user must follow the Getting Started with AWS IoT Greengrass developer guide in order to install and prepare the Greengrass core on the Raspberry Pi. The developer guide must be followed up to at least *Module 3*.

After the Greengrass procedure is completed, the user can continue configuring the Lambda following these steps:

1. The user must make sure the Greengrass core is not active yet. In order to stop the core, the user must open a new terminal window in RPi, go to the location of the core, and perform a Stop command:

- press **CTRL** + **ALT** + **T** to open a new terminal window
- enter '`cd /greengrass/ggc/core`' to open the location of the core
- enter '`sudo ./greengrassd stop`' to stop the core

2. The Lambda and its dependencies are provided in the GitHub repository. The user must open the 'AWS_Lambda' folder and open the '`lambda_function.py`' file with a text editor.
3. The 'DEVICE' variable represents the MAC address of the RN4871 module and must be changed with the one obtained in 4.2.2 Command State. A MAC address has six bytes and the user must add a colon between every byte:
   - `DEVICE = "D8:80:39:F3:75:59"`
4. The user must archive as `.zip` the entire content of the 'AWS_Lambda' folder, and not the folder itself. Then, the user must upload the archive to the cloud and deploy it on Raspberry Pi, as described in the AWS Greengrass tutorial *Module 3*. The topic used when creating the subscription is 'BLE/data'.
5. By default, the RN4871 requires connection with authentication and encryption and the BLE module from RPi does not implement it by default. The name of the device responsible for BLE in RPi will be '`hci0`' and using the '`hciconfig`' command will show the default state of the module:

```
pi@raspberrypi:~ $ sudo hciconfig hci0
hci0:    Type: Primary  Bus: UART
    BD Address: B8:27:EB:09:B1:6A  ACL MTU: 1021:8  SCO MTU: 64:1
    UP RUNNING
    RX bytes:813 acl:0 sco:0 events:53 errors:0
    TX bytes:2524 acl:0 sco:0 commands:53 errors:0
```

   The authentication with encryption must be activated using the '`encrypt`' argument. Using the same command as above for module status checking, new '`AUTH`' and '`ENCRYPT`' functionalities can be observed:

```
pi@raspberrypi:~ $ sudo hciconfig hci0 encrypt
pi@raspberrypi:~ $ sudo hciconfig hci0
hci0:    Type: Primary  Bus: UART
    BD Address: B8:27:EB:09:B1:6A  ACL MTU: 1021:8  SCO MTU: 64:1
    UP RUNNING AUTH ENCRYPT
    RX bytes:827 acl:0 sco:0 events:55 errors:0
    TX bytes:2534 acl:0 sco:0 commands:55 errors:0
```

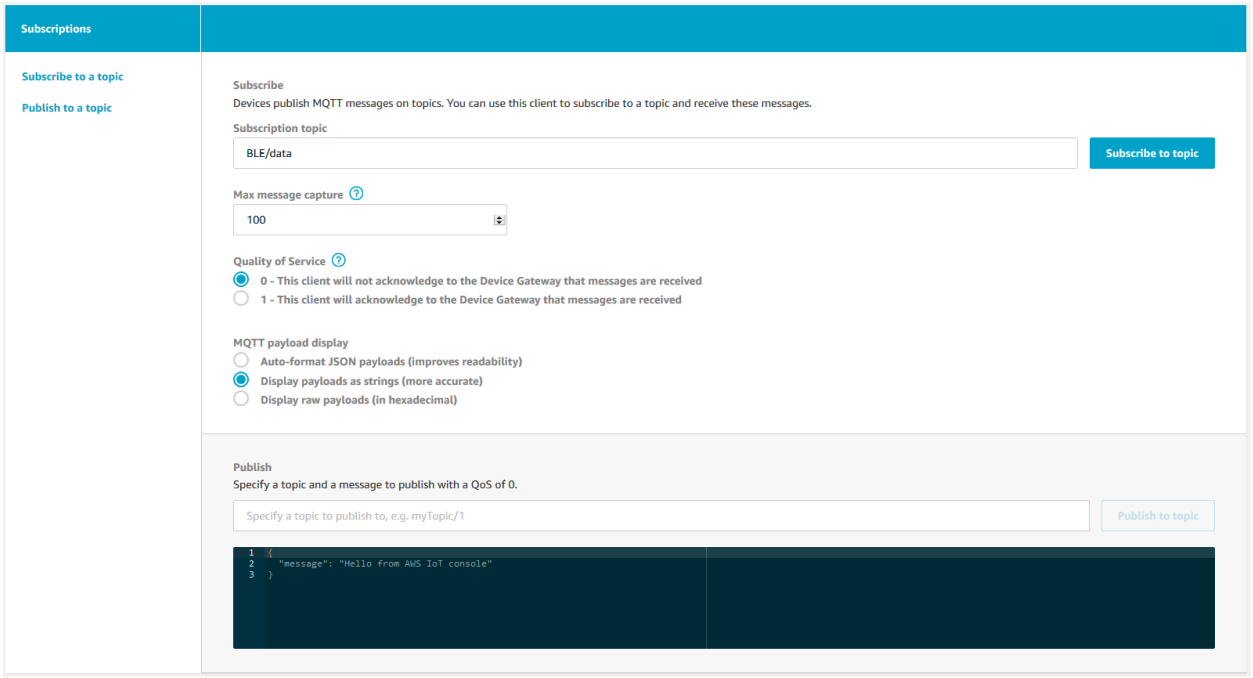
6. All the configurations have been implemented and the Greengrass core can start now:
   - press **CTRL** + **ALT** + **T** to open a new terminal window.
   - enter '`cd /greengrass/ggc/core`' to open the location of the core
   - enter '`sudo ./greengrassd start`' to start the core

   Note: All of the above configurations must be followed only for the first time, when loading a new Lambda. After performing a Reset on the RPi board, only steps 5 and 6 must be followed.

4.4 Visualizing Sensor Data in Cloud

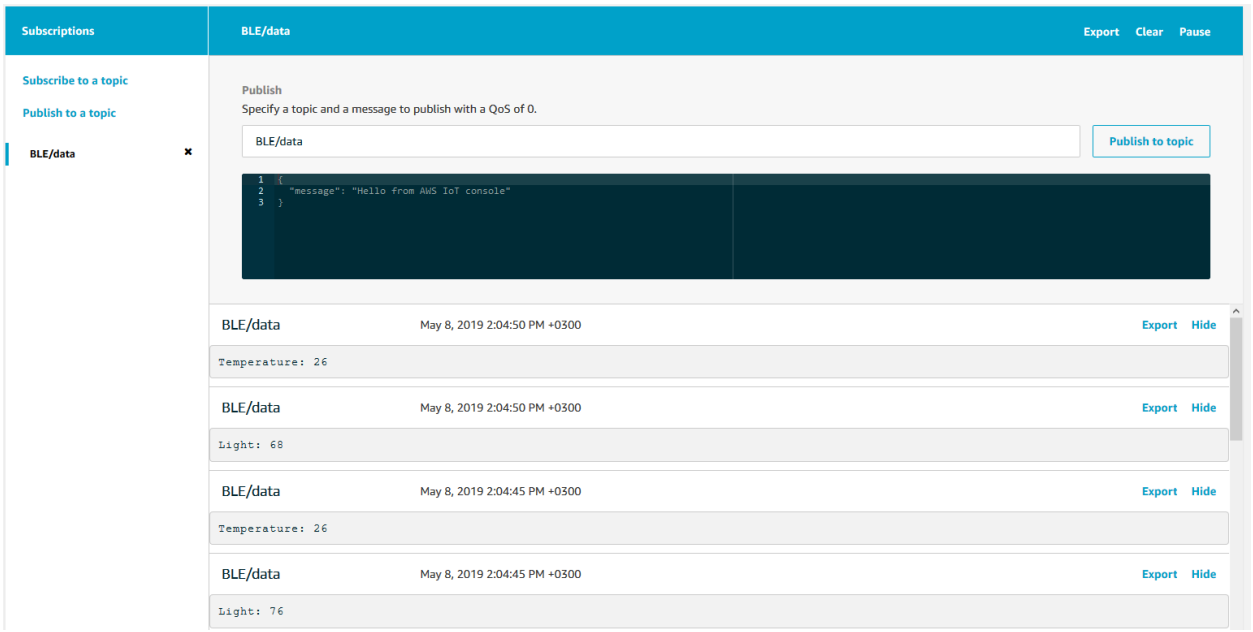
The user must open the personal AWS account and test the subscriptions as presented in *Module 3* of the AWS Greengrass Core Developer Guide. The topic in which the Lambda will publish data is 'BLE/data' and the user must subscribe to it.

Figure 4-2. Cloud Test



After the subscriptions, the window will print the sensors value every five seconds as in the image below.

Figure 4-3. Cloud Data



5. Conclusion

Following the steps presented in this application note, the user can obtain a simple solution of sending BLE data through a gateway to the cloud. Microchip's AVR-IoT WG Development Board provides a mikroBUS socket which allows the users to add new features to their project, such as the RN4871 Click board. Atmel START code generator offers support for this board, configuring the interface and the default settings of the board. The gateway software is provided by AWS together with a developer guide which provides details regarding the configurations and features. The project for the AVR-IoT WG Development Board and the Lambda for the gateway are provided in the GitHub repository.

6. References

1. AVR-IoT WG Development Board
2. AVR-IoT WG Development Board Schematic
3. BLE RN4871 Click board
4. BLE RN4871 Module User's Guide
5. Raspberry Pi 3 Model B+
6. AWS Developer Guide
7. GitHub Repository

The Microchip Web Site

Microchip provides online support via our web site at http://www.microchip.com/. This web site is used as a means to make files and information easily available to customers. Accessible by using your favorite Internet browser, the web site contains the following information:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQ), technical support requests, online discussion groups, Microchip consultant program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

Customer Change Notification Service

Microchip's customer notification service helps keep customers current on Microchip products. Subscribers will receive e-mail notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, access the Microchip web site at http://www.microchip.com/. Under "Support", click on "Customer Change Notification" and follow the registration instructions.

Customer Support

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Field Application Engineer (FAE)
- Technical Support

Customers should contact their distributor, representative or Field Application Engineer (FAE) for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in the back of this document.

Technical support is available through the web site at: http://www.microchip.com/support

Microchip Devices Code Protection Feature

Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.

- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as “unbreakable.”

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip’s code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

Legal Notice

Trademarks

ISBN: 978-1-5224-4524-1

Quality Management System Certified by DNV

ISO/TS 16949

Microchip received ISO/TS-16949:2009 certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona; Gresham, Oregon and design centers in California and India. The Company's quality system processes and procedures are for its PIC® MCUs and dsPIC® DSCs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001:2000 certified.

Worldwide Sales and Service

AMERICAS	ASIA/PACIFIC	ASIA/PACIFIC	EUROPE
Corporate Office 2355 West Chandler Blvd. Chandler, AZ 85224-6199 Tel: 480-792-7200 Fax: 480-792-7277 Technical Support: http://www.microchip.com/support Web Address: http://www.microchip.com **Atlanta** Duluth, GA Tel: 678-957-9614 Fax: 678-957-1455 **Austin, TX** Tel: 512-257-3370 **Boston** Westborough, MA Tel: 774-760-0087 Fax: 774-760-0088 **Chicago** Itasca, IL Tel: 630-285-0071 Fax: 630-285-0075 **Dallas** Addison, TX Tel: 972-818-7423 Fax: 972-818-2924 **Detroit** Novi, MI Tel: 248-848-4000 **Houston, TX** Tel: 281-894-5983 **Indianapolis** Noblesville, IN Tel: 317-773-8323 Fax: 317-773-5453 Tel: 317-536-2380 **Los Angeles** Mission Viejo, CA Tel: 949-462-9523 Fax: 949-462-9608 Tel: 951-273-7800 **Raleigh, NC** Tel: 919-844-7510 **New York, NY** Tel: 631-435-6000 **San Jose, CA** Tel: 408-735-9110 Tel: 408-436-4270 **Canada - Toronto** Tel: 905-695-1980 Fax: 905-695-2078	**Australia - Sydney** Tel: 61-2-9868-6733 **China - Beijing** Tel: 86-10-8569-7000 **China - Chengdu** Tel: 86-28-8665-5511 **China - Chongqing** Tel: 86-23-8980-9588 **China - Dongguan** Tel: 86-769-8702-9880 **China - Guangzhou** Tel: 86-20-8755-8029 **China - Hangzhou** Tel: 86-571-8792-8115 **China - Hong Kong SAR** Tel: 852-2943-5100 **China - Nanjing** Tel: 86-25-8473-2460 **China - Qingdao** Tel: 86-532-8502-7355 **China - Shanghai** Tel: 86-21-3326-8000 **China - Shenyang** Tel: 86-24-2334-2829 **China - Shenzhen** Tel: 86-755-8864-2200 **China - Suzhou** Tel: 86-186-6233-1526 **China - Wuhan** Tel: 86-27-5980-5300 **China - Xian** Tel: 86-29-8833-7252 **China - Xiamen** Tel: 86-592-2388138 **China - Zhuhai** Tel: 86-756-3210040	**India - Bangalore** Tel: 91-80-3090-4444 **India - New Delhi** Tel: 91-11-4160-8631 **India - Pune** Tel: 91-20-4121-0141 **Japan - Osaka** Tel: 81-6-6152-7160 **Japan - Tokyo** Tel: 81-3-6880- 3770 **Korea - Daegu** Tel: 82-53-744-4301 **Korea - Seoul** Tel: 82-2-554-7200 **Malaysia - Kuala Lumpur** Tel: 60-3-7651-7906 **Malaysia - Penang** Tel: 60-4-227-8870 **Philippines - Manila** Tel: 63-2-634-9065 **Singapore** Tel: 65-6334-8870 **Taiwan - Hsin Chu** Tel: 886-3-577-8366 **Taiwan - Kaohsiung** Tel: 886-7-213-7830 **Taiwan - Taipei** Tel: 886-2-2508-8600 **Thailand - Bangkok** Tel: 66-2-694-1351 **Vietnam - Ho Chi Minh** Tel: 84-28-5448-2100	**Austria - Wels** Tel: 43-7242-2244-39 Fax: 43-7242-2244-393 **Denmark - Copenhagen** Tel: 45-4450-2828 Fax: 45-4485-2829 **Finland - Espoo** Tel: 358-9-4520-820 **France - Paris** Tel: 33-1-69-53-63-20 Fax: 33-1-69-30-90-79 **Germany - Garching** Tel: 49-8931-9700 **Germany - Haan** Tel: 49-2129-3766400 **Germany - Heilbronn** Tel: 49-7131-72400 **Germany - Karlsruhe** Tel: 49-721-625370 **Germany - Munich** Tel: 49-89-627-144-0 Fax: 49-89-627-144-44 **Germany - Rosenheim** Tel: 49-8031-354-560 **Israel - Ra'anana** Tel: 972-9-744-7705 **Italy - Milan** Tel: 39-0331-742611 Fax: 39-0331-466781 **Italy - Padova** Tel: 39-049-7625286 **Netherlands - Drunen** Tel: 31-416-690399 Fax: 31-416-690340 **Norway - Trondheim** Tel: 47-72884388 **Poland - Warsaw** Tel: 48-22-3325737 **Romania - Bucharest** Tel: 40-21-407-87-50 **Spain - Madrid** Tel: 34-91-708-08-90 Fax: 34-91-708-08-91 **Sweden - Gothenberg** Tel: 46-31-704-60-40 **Sweden - Stockholm** Tel: 46-8-5090-4654 **UK - Wokingham** Tel: 44-118-921-5800 Fax: 44-118-921-5820