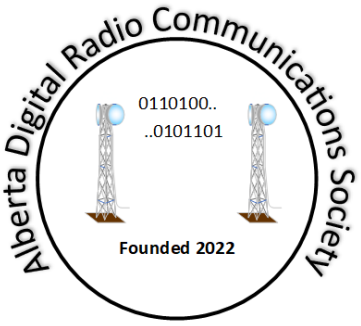

APRS over MQTT Platform

Arduino R4 AQI Sensor Project Documentation

Document Number: TBD
Revision: 0.1
Status: Preliminary

Table of Contents

List of Tables

List of Figures

References

[1] gnu.org, "General Public Licence," [Online]. Available: https://www.gnu.org/licenses/gpl-3.0.en.html. [Accessed 25th February 2018].

[2] Wikipedia, "Automatic Packet Reporting System," [Online]. Available: https://en.wikipedia.org/wiki/Automatic_Packet_Reporting_System. [Accessed 22 August 2024].

[3] Wikipedia, "Internet of Things," [Online]. Available: https://en.wikipedia.org/wiki/Internet_of_things. [Accessed 22 August 2024].

[4] MQTT, "MQTT: The Standard for IoT Messaging," [Online]. Available: https://mqtt.org/. [Accessed 22 August 2024].

[5] Bosch Sensortech, "BME680 Low power gas, pressure, temperature and humidity sensor," [Online]. Available: chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://www.bosch-sensortec.com/media/boschsensortec/downloads/datasheets/bst-bme680-ds001.pdf. [Accessed 22 August 2024].

Revision Status

Revision	Date	Description
0.1	October 1, 2024	Initial draft

Table 1 Revision status

Intellectual Property Notice

The hardware components and all intellectual property described herein is the exclusive property of the Alberta Digital Radio Communications Society and others (“the owner”), all rights are reserved.

The owner grants licence to any Amateur for personal or club use, on an as is and where is basis under the condition that its use is for non-commercial activities only, all other usages are strictly prohibited. Terms and conditions are governed by the GNU public licence [1].

No warranty, either express or implied or transfer of rights is granted in this licence and the owner is not liable for any outcome whatsoever arising from such usage.

Disclaimer

This document is a preliminary release for a product still in development and may be subject to change in future revisions. The software contained herein may be subject to unpredictable behaviour without notice. You are advised to keep a can of RAID™ Ant, Roach and Program Bug killer handy. Spray liberally on the affected area when needed.

Introduction

The Automatic Packet Reporting System (APRS) was introduced in the late 1980s by Bob Bruninga, WB4APR [2], a senior researcher at the US Naval academy, and has since grown to a world-wide network that reports position information as well as weather conditions and telemetry. It has become a tool used by amateurs worldwide for data acquisition and monitoring.

APRS relies on amateur frequencies and a digital packet format using audio frequency shift keying (AFSK) at 1200 bits/second. Data can originate at a mobile or fixed station, and to facilitate reporting to the APRS database, two types of receiving stations have been deployed, those that simply repeat what they hear to extend communications range (digipeaters), and those that provide a bridge to the commercial internet for reporting purposes (I-Gates). Access to the database is limited to the deployment of these station types, and their coverage is not ubiquitous.

Similarly, in the commercial arena, more and more devices are utilizing a technology commonly referred to as the 'Internet of Things' (IOT) [3] which acts in a similar manner. This has let to the development of a new service to support both stationary and mobile stations, and the emergence of an internet protocol specifically designed for this purpose, known as the Message Queuing Telemetry Transport, or MQTT [4]. As this is supported by commercial ventures, the coverage is extensive.

The objective of this project is to marry these two technologies together and create a platform that can bridge data sent over MQTT and post it to APRS. This not only opens up a global coverage area but enables new devices to be able to source data and add new reporting applications.

This particular application measures the Air Quality Index (AQI), using a Bosch BME680 sensor, and an Arduino R4 WiFi development board. The sensor measures, temperature, pressure, relative humidity, and calculates the AQI.

Indoor Air Quality Index

The AQI is an integer ranging from 0 to 500, divided into 6 sub-ranges of 50 each. The calculation is made based on Volatile Organic Compounds (VOC's) in the surrounding air.

Figure 1 illustrates the index values and their meaning.

IAQ Index	Air Quality
0 – 50	good[10]
51 – 100	average
101 – 150	little bad
151 – 200	bad
201 – 300	worse[2]
301 – 500	very bad

Figure 1 Aiq Quality Index values and meaning

Project Set up

Two hardware components are required, an Arduino Rev 4 Wi-Fi board, and a Wave Share BME680 [5] sensor board. Both can be purchased locally for about $60.

Connecting the Sensor Board

Connect the Sensor board to the R4 as illustrated in Figure 2

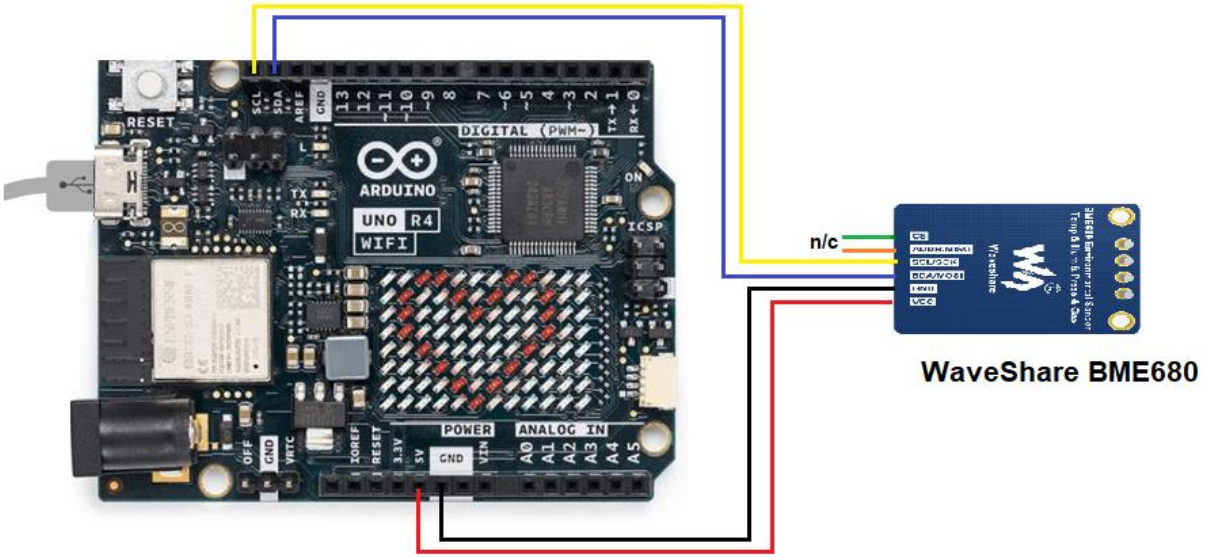

Figure 2 Connecting the Sensor Board

There are six wires from the sensor board, the green and orange are not connected. The interface runs in I2C mode, which only requires the wires shown, and power and ground.

Upgrading the firmware

WARNING From Paulo, VE8PR: I advise against trying to update the UNO R4 WIFI firmware using Firmware Updater, I tried to do this and corrupted it which took me hours to find a solution to reinstall the new firmware.

This has been confirmed as well. An out-of-the-box device worked correctly, but it failed after an upgrade. It is advisable not to attempt any upgrades currently.

Setting up the Development Software

Ensure that you have a recent version of the Arduino IDE installed. Ensure that the all the libraries in the library sub-directory are installed.

Library	Purpose
ArduinoGraphics	Enables using the LED matrix as
ArduinoMQTTClient	Client library for the MQTT protocol
BME68x_Sensor_Library	Base functions for the BME680. Required.
Bsec2	Update to the basic library.
PubSubClient	MQTT publish and subscribe library

Table 2 Required Libraries

Configuring the application

Before compiling the application, several parameters need to be setup up, some of which are location specific. First, open the file "config.h" and set the following:

Parameter	Purpose	Default
CONFIG_NETWORKID	ID and password of your Wifi Network. Dependent on you installation.	
CONFIG_PASS		
CONFIG_BROKER	Address of MQTT Broker	aprs.adrcs.org
CONFIG_PORT	Port number	7000
CONFIG_MQTT_USER	Username for MQTT login	User dependent
CONFIG_MQTT_PASS	Password for MQTT login	
CONFIG_DEVICE	Device ID. A unique string to identify your device. A suggestion is to start with your phone number and add additional digits on the end.	
CONFIG_GPS_LAT	Your latitude and longitude. Up to 6 digits can be specified, the best source is to right click on your location in google maps.	
CONFIG_GPS_LONG		

Table 3 Application configuration

In addition, there are three debug options that can be set, set them to 1 to enable and zero to disable.

Option	Effect
DUMP_PAYLOAD	Dumps the payload to the serial monitor when sending an MQTT packet
SHORT_FUSE	Shortens the interval between posts to 5 minutes instead of 15
WIFI_STATS	Dumps the WiFi stats to the serial monitor when connected

Table 4 Debug options

Compiling

Launch the Arduino IDE and compile the sketch to expand the libraries. Using a file browser, enter the expanded directory for the 'ArduinoMqttClient' library, the then subdirectory 'src'. Open the file MqttClient.h with a text editor.

Add the following line to the file right after the #include directives at line 24:

```
#define TX_PAYLOAD_BUFFER_SIZE              700
```

Write the file, close it, relaunch the IDE and recompile the sketch. Download it to the board and it is ready for use.

Setup

Prior to running the software, launch a mesh-enabled web browser, and go to the PNW Information site, listed on node VE6VH-hAP-AREDN. At the bottom of the home page, click the link to 'ADRCS Site', then click on the "APRS over MQTT Site" link in the menu bar.

Click the 'Log In' from the menu bar, then if you do not have an account, create one first, then log into it. Click the 'Devices' link on the menu bar, and if none are listed, then the link to add a new device. The form shown in Figure 3. Enter the same device ID as you used in the **CONFIG_DEVICE** parameter entry. The description is not used but can be anything to describe the device.

Figure 3 New Device form

The APRS Callsign should be your own, and the SSID is numeric, from 0 to 15. Click submit to enter it into the database, then the 'Devices' link again at the top of the page. You should see your device there.

You are now ready to run the application.

Operation

Connect the serial monitor in the IDE at 9600 bps, or use PuTTy to connect when the IDE is closed. The software first connects to the local WiFi, then to the MQTT broker. This connection is maintained the entire time the software is operational.

It will wait for the sensor to become ready, then transmit a packet right away. The packet should appear on aprs.fi under the callsign and SSID you entered at the website. After the first packet, the application waits for up to 15 minutes (depending on the **SHORT_FUSE** setting) , and it will display a countdown in the LED's while waiting.

When ready to send another reading, it can take a few minutes before it is ready. An announcement is made on the serial monitor when it is waiting or sending.

Enjoy.