LoRa Module Low Power Testing Report

By: Bob Glicksman; 5/04/24

OVERVIEW.

This document describes the low power mode tests performed on the RYLR998 LoRa modules (https://reyax.com/products/RYLR998). These tests were performed in order to verify that these modules can be powered down to under 10 microamps when not being used to transmit data. Specifically, these tests verified that the modules can be powered up, then transmit a short message and then be powered down. This operation is necessary for a battery powered sensor, such as needed for the IVGM project.

Figure 1. RYLR998 LoRa Modem Module.

RESULTS.

The RYLR998 LoRa modules were measured to draw about 17.5 milliamps when receiving (mode = 0) and about 8.5 microamps when in low power mode (mode = 1). When switching from mode 0 to mode 1, the module returns "+OK/r/n" and then powers down. When powered down, switching back to mode 0 returns "+OK/r/n" and the module powers up. Note that when in low power mode, sending a message (e.g. "AT+SEND=1,5,HELLO") powers the module back up but returns "+MODE=0" as opposed to "+OK". We think it best to explicitly power the LoRa module up before sending a message.

TESTING METHODOLOGY.

Mode and Power Testing.

An FTDI module was directly wired to an RYLR998 module as follows:

- FTDI GND to RYLR998 GND
- FTDI +3.3 to ammeter + (multimeter); ammeter – to RYLR998 VDD
- FTDI Tx to RYLR998 RXD
- FTDI Rx to RYLR998 TXD
- FTDI +5 unconnected
- RYLR998 NRST unconnected

The Arduino IDE Serial Monitor was used for communication. The current draw from the +3.3 volt power was measured by the multimeter on the 20 ma scale and also on the 200 microamp scale (lowest current scale available). The Serial Monitor was set to the RYLR998 default baud rate of 115200 and the end line set to both CR and LF, per the RYLR998 documentation; see:

https://reyax.com/products/RYLR998

Scroll down to “AT COMMAND MANUAL”.

The command “AT+MODE?” returned “+MODE=0”, which is consistent with the manufacturer’s documentation. The command “AT+SEND=1,5,HELLO” sent out this data, but the transmission was too quick to measure current draw on the ammeter. However, the ammeter did respond briefly, indicating that the current draw went way up when transmitting, consistent with the manufacturer’s data sheet.

The command “AT+MODE=1” returned “+OK/r/n” and the module powered down to 8.5 microamps. The command “AT+MODE=0” returned “+OK/r/n” and the module powered back up to 17.5 milliamps.

The command “AT+MODE=1” returned “+OK/r/n” and the module powered down to 8.5 microamps. The command “AT+SEND=1,5,HELLO” appeared to power the module back up and send out the message, but it returned “+MODE=0/r/n” in lieu of “+OK/r/n”.

These tests verify that the module powers up and powers down according to the manufacturer’s data.

Communication Testing with Power Switching.

These tests were performed to confirm the timing operation of powering up the LoRa module, sending out a short message, and then powering the module back down. It was performed in order to confirm that a command is complete and a new command to the module can be sent immediately after "+OK/r/n" is returned from the module to the microcontroller. In particular, we confirmed that we can power the LoRa module down immediately upon receiving "+OK/r/n" after a SEND command and that the LoRa transmission will be complete.

Two LoRa modules were pre-configured, as described below.

Figure 2. Range Testing Architecture.

The module designated for the "tester" was connected to a Particle Photon microcontroller that was programmed to send a short test message via the LoRa module whenever an on-board pushbutton was pressed. Specifically, the LoRa module is powered up, the test message is sent to the hub, and the LoRa module is powered down. The response message from the module ("+OK/r/n") must always be received before a new command can be issued, but no further delays are required. Powering the LoRa module down was confirmed with an ammeter on the 3.3 volt power to the module.

The module designated for the "hub" is the same as the hub used for range testing. The hub's LoRa module was connected to a Particle Photon microcontroller that was programmed to wait for receipt of a test message from the "tester" and, if a valid message was received, to transmit back a response message to the "tester" via its own LoRa module. The received message is also sent to a host computer via the Photon's USB serial

port and the message is displayed on a serial monitor for confirmation that the full and correct message was received.

Both the “tester” and the “Hub” were powered via USB on the Photon.

EQUIPMENT DESCIPTION.

Parts.

The following parts were used for these tests:

- 2 ea. RYLR998 LoRa Modem Modules. https://www.amazon.com/dp/B099RM1XMG?psc=1&ref=ppx_yo2ov_dt_b_product_details
- 1 ea. USB-Serial (“FTDI”) Module. https://www.amazon.com/dp/B00LODGRV8?psc=1&ref=ppx_yo2ov_dt_b_product_details
- 2 ea. Particle Photon modules. https://www.particle.io/. NOTE: The Photon is currently out of production; the Photon 2 may be used instead. The WiFi and BLE capabilities of the Photon/Photon 2 are not used for these tests and any 3.3 volt/USB powered Arduino board may be used.
- 1 ea. Breadboard compatible pushbutton switch.
- 2 ea. Solderless breadboard and male-male jumper wires.
- 1 ea. USB battery power supply.
- 1 ea. Multimeter.

Construction.

The figure below shows the “tester”:

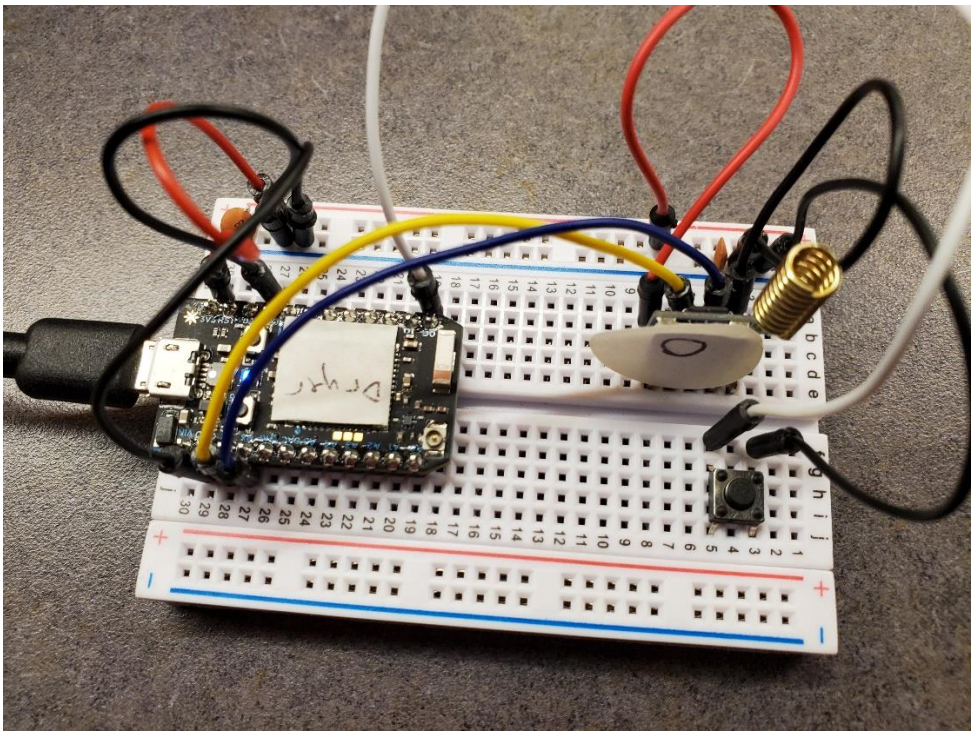

Figure 4. Photo of the Tester.

The following figure shows the wiring of the “tester”:

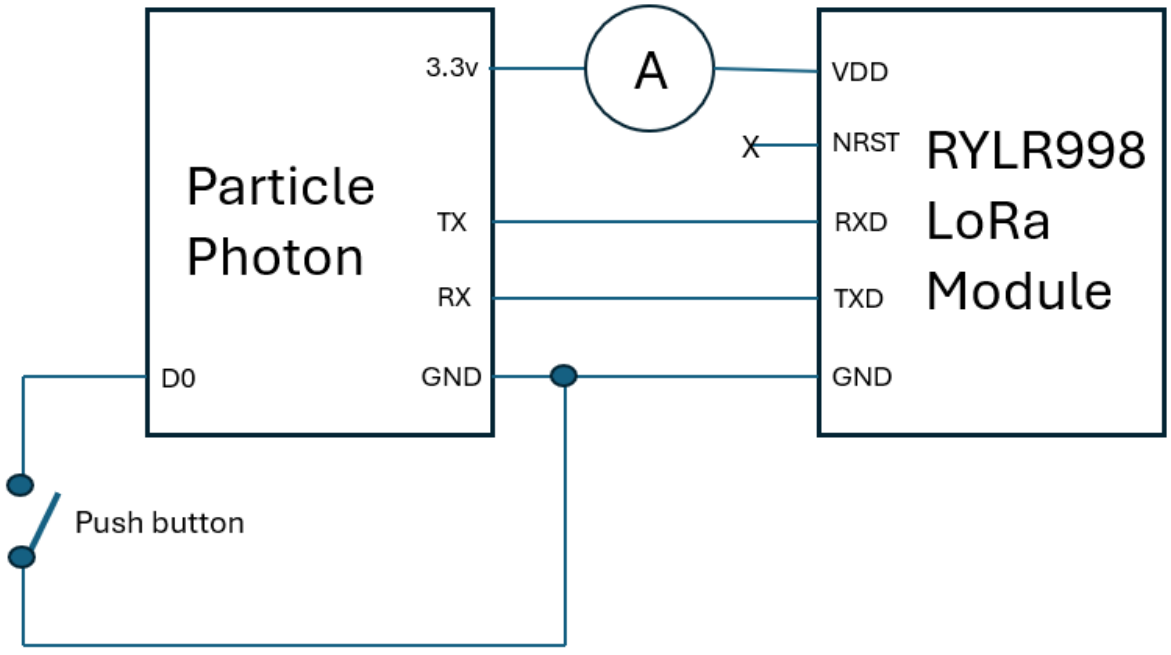

Figure 5. Tester schematic.

The Photon is powered through its onboard USB port. The Photon's 3.3 volt output (from its on-board regulator) is used to power the LoRa module though an ammeter (multimeter on the DC amps setting). The Photon's TX pin (Serial1 port) is connected to the LoRa module's RXD pin and visa-vera. The LoRa module's NRST (reset) pin is not connected.

The pushbutton switch has one end connected to the Photon D0 pin and the other end connected to ground.

The figure below shows the "hub":

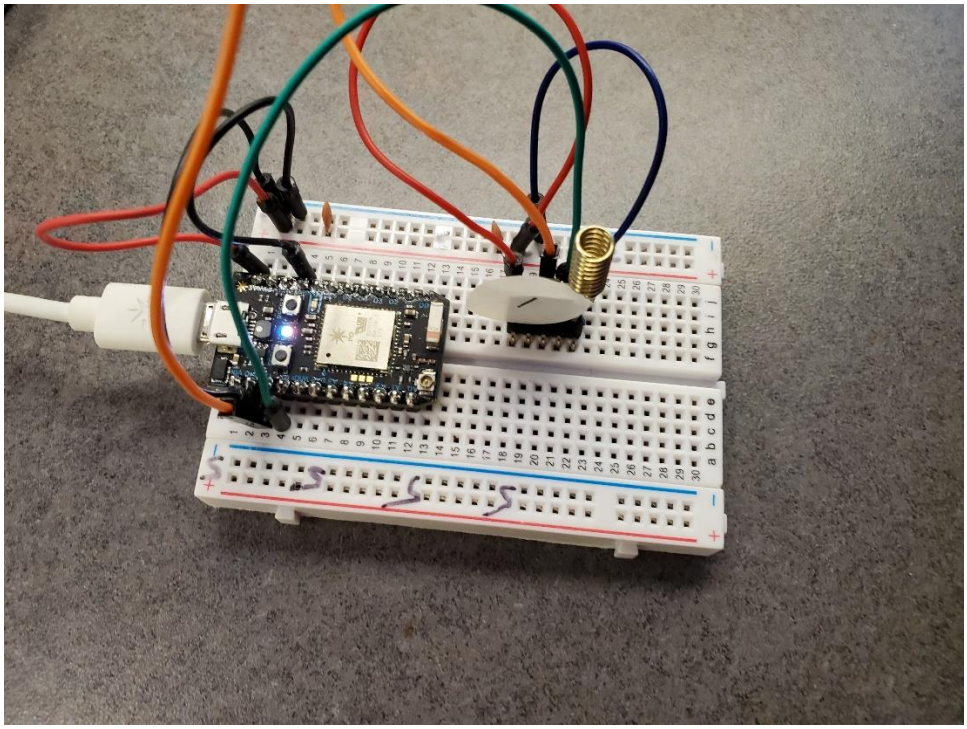

Figure 6. Photo of the Hub.

The following figure shows the wiring of the "hub":

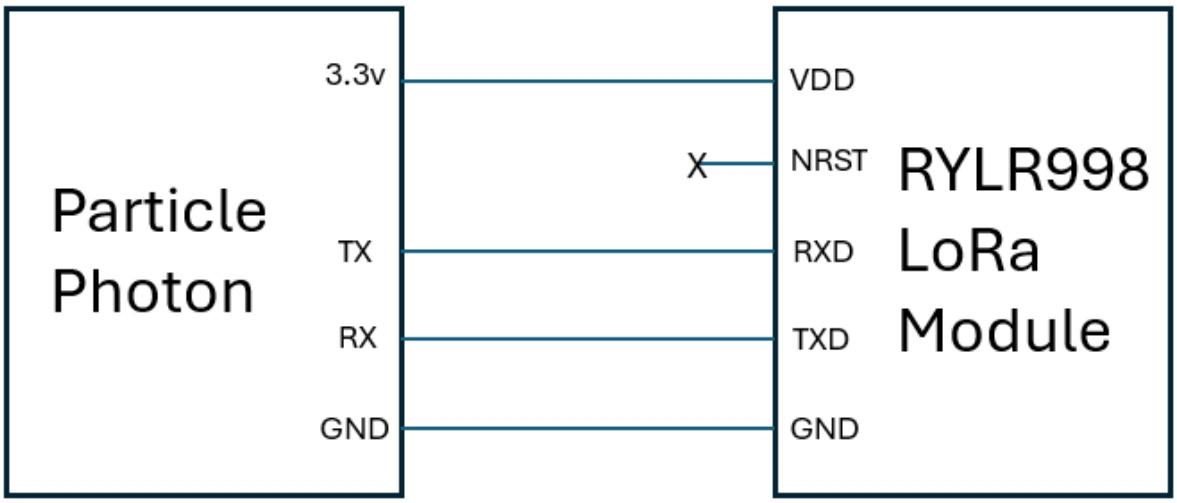

Figure 7. Hub Schematic.

The “hub” is the same (hardware and software) as for LoRa Range Testing.

SETUP AND CONFIGURATION.

The “hub” and “tester” software does not configure the LoRa modules. The modules must be set up using the USB-Serial module (“FTDI” module) and a PC running a terminal program. The terminal program used for these tests was the Arduino serial monitor; however, any terminal program will work.

The LoRa modules come “out-of-the-box” with a number of default settings. In general, the default settings were used for these tests. However, the module number and the network number were changed. The official documentation of the module AT command set can be found at:

https://reyax.com/products/RYLR998

Scroll down to “AT COMMAND MANUAL”.

The default baud rate for these LoRa modules is 115200 baud, and the default end-of-line is CR and LF (“/r/n”). In order to communicate with these modules using a terminal program on a PC, this baud rate and end-of-line settings must be used. Here are the settings used for these tests:

Hub:

- Baud rate: 115200 (default)
- Mode: 0 (default)

- Band (for USA): 915 MHz (default)
- Network ID: 3
- Address: 0 (default)
- Parameters: (all default)
    - Spreading factor: 9
    - Bandwidth: 125 KHz
    - Coding Rate: 1
    - Preamble: 12

Tester:

- Baud rate: 115200 (default)
- Mode: 0 (default)
- Band (for USA): 915 MHz (default)
- Network ID: 3
- Address: 1
- Parameters: (all default)
    - Spreading factor: 9
    - Bandwidth: 125 KHz
    - Coding Rate: 1
    - Preamble: 12

Note that only the network ID and the module address were changed from the defaults. The two modules must be set to different addresses, but they must both have the same network ID.

SOFTWARE DESCRIPTION.

The software (firmware) source code for the “hub” and “tester” is included in this repository. The software is well commented; hence only a brief description is given here.

Tester Software.

The “tester” software must operate independently of whether the “tester” can connect to WiFi. In order to use a Particle Photon, the “semi-automatic” system mode must be declared up front.

The “tester” software sets up things in setup(), including setting up D0 as INPUT_PULLUP and setting the Serial1 (LoRa module) baud rate for 115200. In loop(), the software tests for depression of the pushbutton. When the button is depressed, it :

1. Powers up the LoRa module (“AT+MODE=0”), and waits for the full OK message to be received.
2. Sends the “hello” message to the hub (“AT+SEND=1,5,HELLO”), and waits for the full OK message to be received.
3. Powers down the LoRa module (“AT+MODE=1”), and waits for the full OK message to be received.
4. Waits for the push button to be released, and then waits for it to again be depressed.

NOTE: The “+OK” must be received from the LoRa module before a new command can be sent (according to the manufacturer’s documentation). We are not explicitly testing for “+OK/r/n”) because what would we do if we got an error instead? Rather, we simply wait for a complete response message from the LoRa module after each command is sent. A complete response is assured by setting *Serial1.setTimeout(10);* in setup(). With this setting, *Serial1.readString();* keeps reading characters from the serial buffer and adding them to the received data string until there are no more characters for 10 milliseconds. This timeout should be more then long enough to assure that a complete response string has been received, event at a lower serial baud rate e.g. 9600 baud).

Hub Software.

The “hub” software is the same as for LoRa range testing. The “hub” software sets up things in setup(), setting the Serial1 (LoRa module) baud rate for 115200, and the Serial (USB) baud rate to 9600 (for debugging using a terminal program), and then enters loop(). In loop(), the software tests for data in the Serial1 receive buffer. If there is data in the buffer, is it read out and appended to a string variable to assemble the received message. Each time that data is added to the received message string variable, it is tested to see if the message is the test message from the “tester” (“HELLO”). If the test message has been received, the “hub” prints the received message to the USB serial port, sends out the response message back to the “tester”, and clears out the message string variable for another test.