Wireless Sensor
Manual
V0.9

Contents

Introduction

Easy to use wireless data transmission modules where messages all the error checking, encoding, packetisation and CRC done for you. Build prototypes in minutes. Require no programming and no drivers. Long range communication up to 1 KM. Support point to multipoint, multipoint to point, multipoint to multipoint or point to point networks. All devices have built in 128-bit AES encryption for secure over the air transmissions.

The devices are configurable through the serial interface or over the air.

Introduction to RF Networks

Star Network

Any number of nodes transmitting and receiving data from one central hub. Sensor nodes can sleep to conserve battery power. No direct communication between end devices.

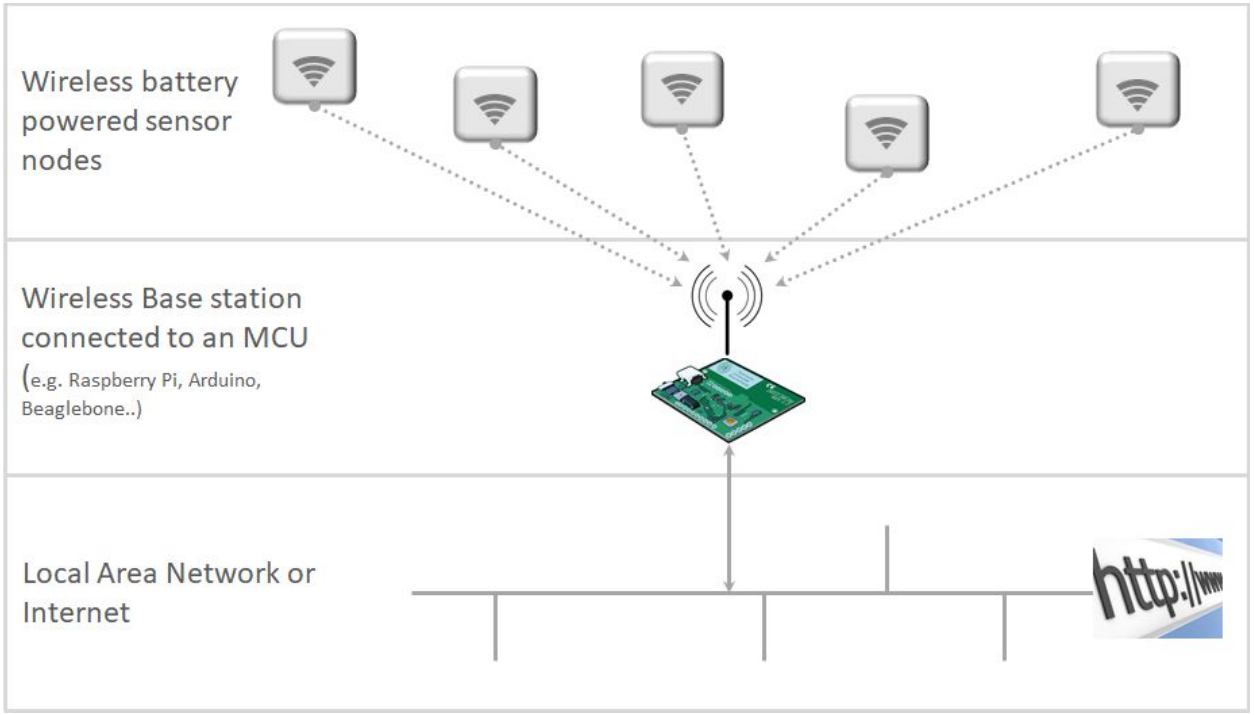


Figure 1 - Star network design

Multiple Star Networks

Start networks can be combined by creating multiple star networks on their own PanId's. Messages can be directed to specific gateways by assigning PanId's to each sensor.

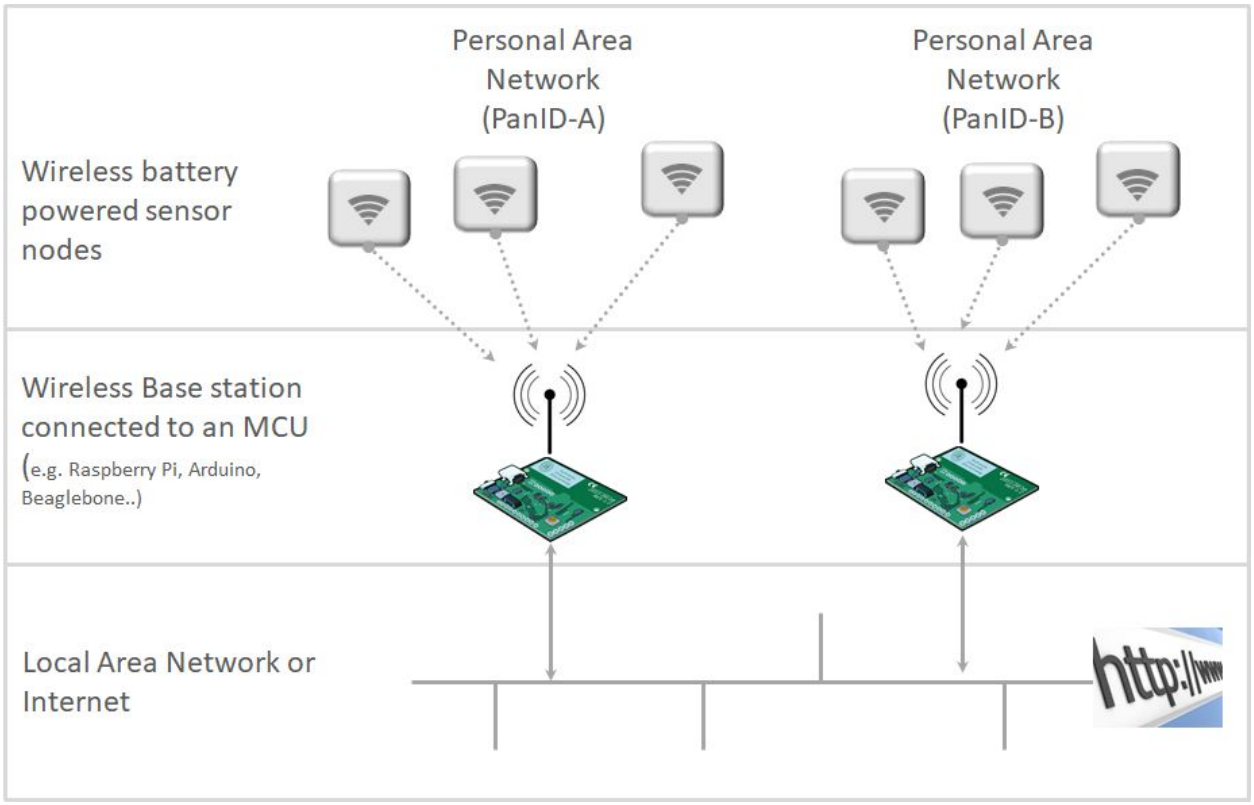


Figure 2 - Multiple star network design

Redundant network

Multiple gateways can be configured on the same PanId to create a redundant network. All radio traffic travels to all gateways so if you lose a gateway then you have a hot backup. Additional gateways can also be installed in areas of poor reception to improve network coverage. Message deduplication logic must be built either within the gateway or further down in back end systems connected to the network.

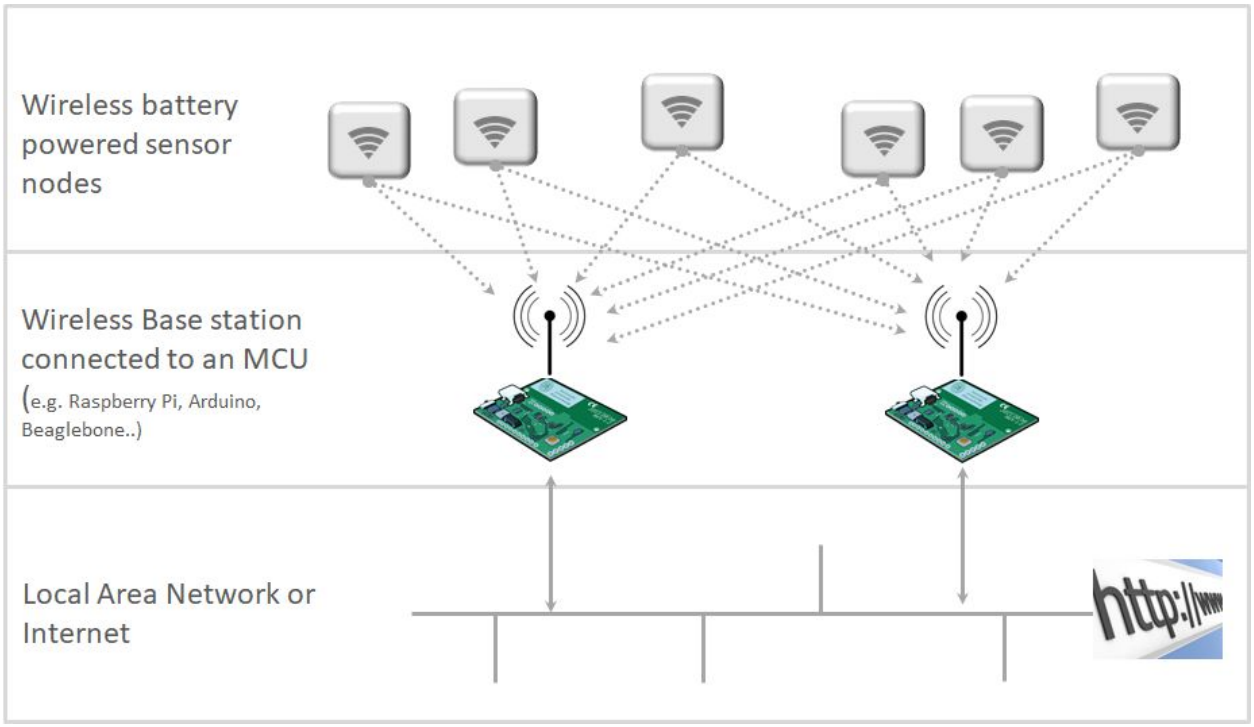


Figure 3 - Redundant network design

Point-to-point Communications

Two or more MCU's can communicate directly with each other as shown in figure 4.

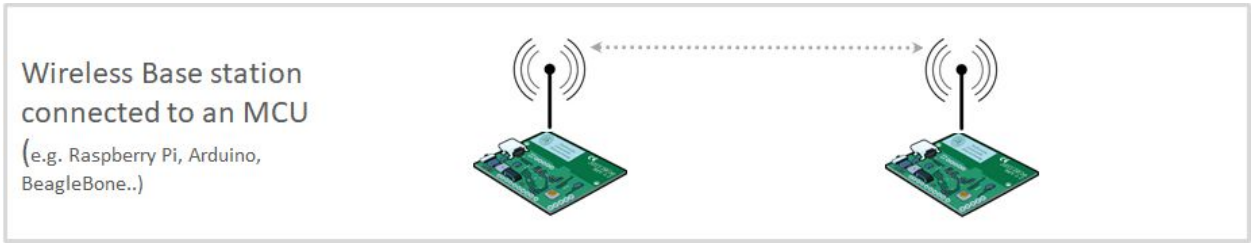


Figure 4 - Point-to-point network

Wireless devices

Tech Specs

- All devices are based on the high-performance RF transceiver based on the market-leading CC1100 SOC
- Wide operating voltage range of 2-3.6V makes the devices suited for battery power
- Current consumption: RX: 16.2 mA , TX: 15.2 mA. Deep Sleep : 0.005Ma (0.5 µA)
- Up to 1KM line of sight. Short range (30M) penetration of walls and floors.
- 128-bit AES encryption security

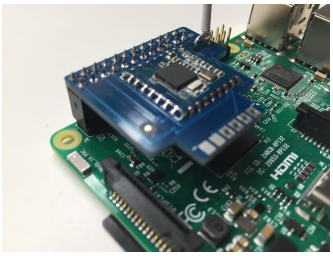
Features

- Driverless installation
- Gatweway communicates through serial port (TTL UART)
- Communicates in ASCII clear text making it very easy to exchange data between devices over the air
- Supports point to point, point to multipoint, multipoint to point and multipoint to multipoint
- Virtually unlimited amount of devices can be deployed. 99,999 Personal Area Network ID’s, 10 channels per frequency and 7,744 Device ID’s per network.
- Super low power consumption allows devices to be use with a coin cell battery for long periods of time (up to 1 year depending on transmission rates).
- Supports 6 frequencies (433 MHZ, 915 MHZ (default US & Canada), 868.3 MHZ (default Europe), 868 MHZ, 903 MHZ, 315 MHZ)
- 10 channels
- 4 pins for communicating with external microcontrollers like Raspberry Pi and Arduino : Tx, Rx, GND, 3V3
- Battery monitor to keep track of power consumption

Introduction to the wireless devices

Gateway Receiver for Raspberry Pi

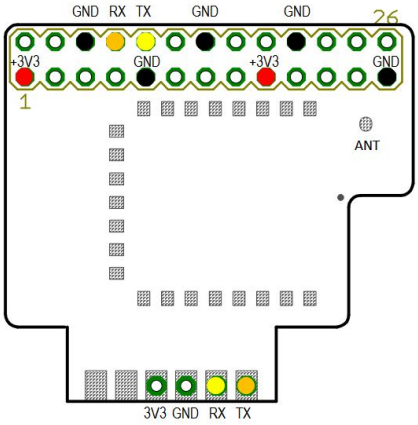
The Gateway for Raspberry Pi is a radio/serial converter. The device communicates with the Raspberry Pi through the serial port. It is so low profile that it will fit inside a range of Pi cases such as the PiBow and Stealth. Compatible with all models of Raspberry Pi.

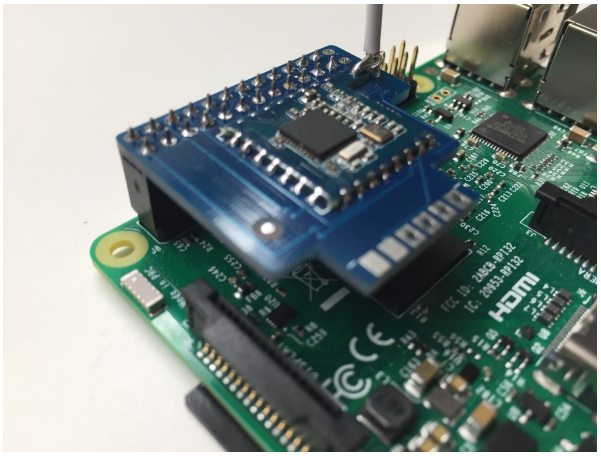


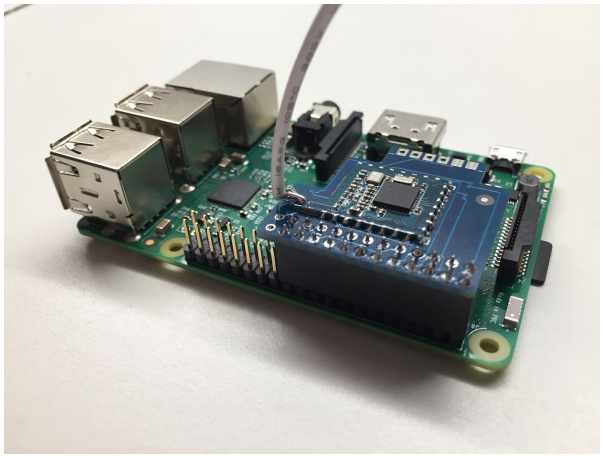
Installation Steps

1. Power down the Raspberry Pi
2. Attach the Gateway Receiver to the GPIO header on the Raspberry Pi as shown in the pictures below.

Caution!!! Be very careful not to misalign the header of the Gateway Receiver as this can damage the device if powered on whilst it is connected to the wrong pins.

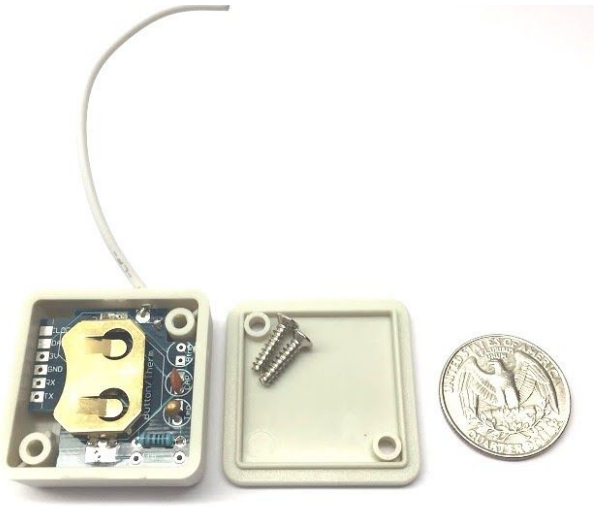






Low Power Wireless Sensor Node

The low power wireless sensor node provides secure, wireless data transmission of sensor in a simple, low cost package. Designed to work with coin cell battery power in deep sleep mode which can last for up to 1 year depending on data transmission rates. Rugged but elegant cream plastic plastic case allows the sensor to be deployed both indoors and outdoors and if sealed with silicone is water resistant. Using an external power source it can operate in request/reply or publish mode. When asleep it is awoken periodically to send readings. Dimensions of case is 2.54cm x 2.54cm (1” x 1”).



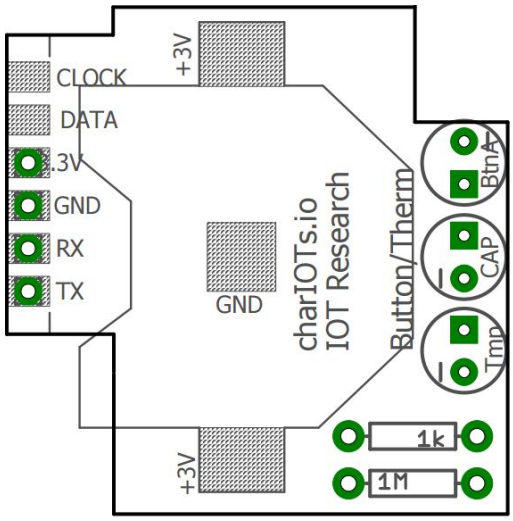
Legend:

BtnA : Terminals to connect any sensor that opens/closes a circuit:

- Push button (see battery restrictions)
- Reed switch
- Magnetic door/window sensor
- Motion sensor
- Water/moisture sensor

Tmp : 10k Thermistor for temperature readings

Board Layout:



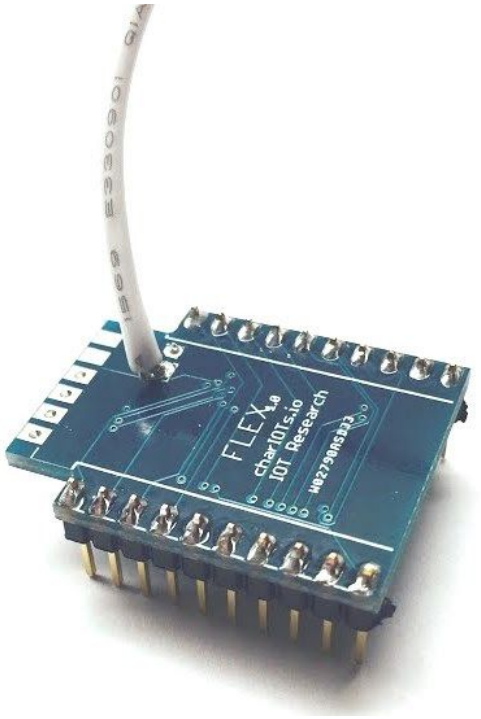
Flex Module

Versatile radio module that can be used as a sensor node or Gateway. Can be used for quick prototyping on a breadboard or part of a permanent solution fixed to a PCB.

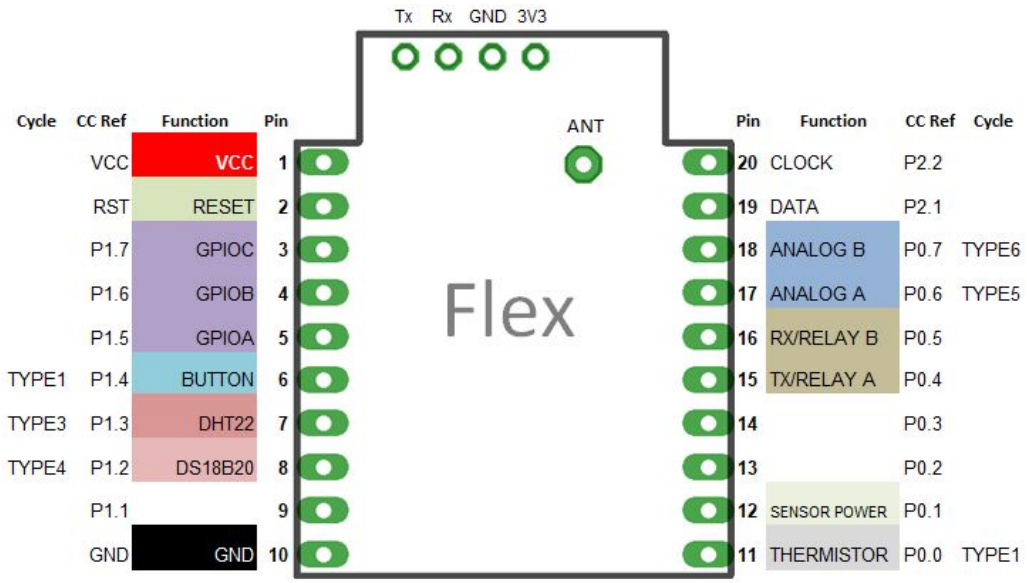
20 pins provides access to all available functionality. The radio module comes with support for a multitude of external sensors often used for Internet Of Things (IOT) projects.

All functionality is built into a single firmware and can be accessed without having to update firmware to a different version:

- UART serial port that enables communications with other Micro-controllers (e.g. Raspberry Pi, Arduino, BeagleBone etc..)
- Thermistor temperature sensor
- DS18B20 digital temperature sensor
- DHT11 digital temperature and humidity sensor
- 2 x Analog inputs
- Button sensor (see battery restrictions)
- 2 x Relay switch controllers
- 3 GPIO pins that can be switched on/off



Flex Pinout Reference



Legend:

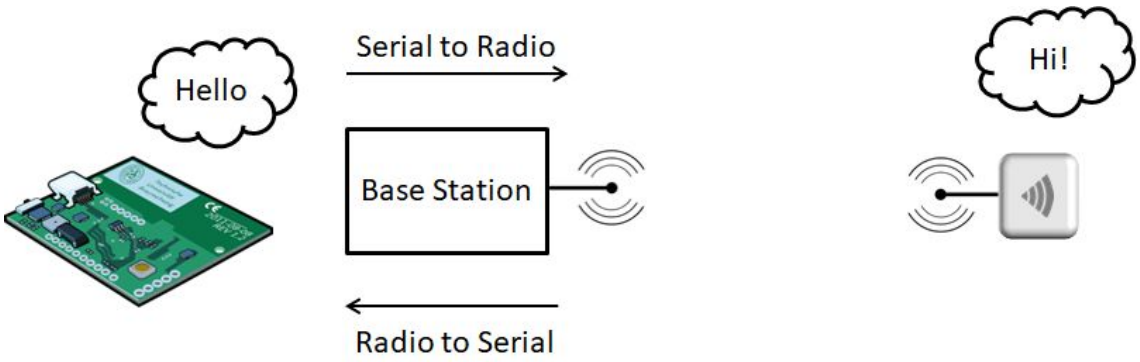
Cycle
Indicates the sensor (refer message reference for TYPE) that the sensor will send values for when put into a sleep cycle (refer message reference ot CYCLE). E.g. if a sensor is set as TYPE4 and put into a sleep cycle, the sensor will transmit temperature readings from the DS18B20 sensor.

CC Ref
This is the mapping to the CC1100 chip. This can be used if you want to reprogram the device firmware (outside the scope of this manual). Pins 1, 10, 2, 19 and 20 (3V3, GND, RESET, DATA, CLOCK) can be use to connect the sensor to a Texas Instrument CC Debugger device for loading custom firmware.

Function
The function provides the pin reference for external devices. The various functions are explained in the next section.

Pin
This is the Flex Pin number used to reference the pins on the Flex radio module.

RF Communication Basics



Following the above diagram the microcontroller (MCU) communicates with the Gateway through the serial port and the Gateway converts serial to radio and radio back to serial. The Gateway wants to send a “Hello” message to a sensor node. The Microcontroller sends “Hello” to the serial port (Tx) and the Gateway receives the message through it’s serial port (Rx), then the Gateway transmits the “Hello” message and the Sensor Node, on the right, receives the radio transmission and replies “Hi!”, which is received by the Gateway and sent back to the Microcontroller through the serial port.

All the radio modules use a two character device id tag to identify themselves. A Gateway will process all messages coming in over the air and pass them to the MCU with the device ID so that the MCU can process accordingly. When a radio module receives a message that has a device id that matches its device id it will processes the message and transmit a reply over the medium (radio or serial) it received the message. If a Gateway receives a message that is not the same is its ID then it passes the message through (either radio to serial or serial to radio).

As shown in the above diagram, messages can be received and sent either through a serial port (in the case of the Gateway) or over the air. Messages are processed in the same manner regardless of whether they arrived through the serial interface or over the air.

Here are some typical examples:

Gateway Device ID : 01
Sensor Node Device ID : 10

1. Sensor node sends a temperature reading to the Gateway every 5 minutes.
    - Sensor node awakens from sleep by the internal timer
    - Sensor node transmits a temperature reading
    - Gateway receives the temperature reading
    - The Device ID of the message is not equal to the Device ID of the Gateway and therefore the Gateway sends the data to the serial port for processing by the MCU

2. Base station sends a configuration change to a sensor node that is awake
    - MCU sends the message to the Gateway over the serial port with an ID of 10
    - Gateway receives the message and it is not the same as its ID and therefore transmits the message (serial to radio) with an ID of 10 asking the device to change a configuration (refer message reference section for all configuration options).
    - The sensor node receives the message with an ID that is the same as its ID and therefore processes the message and sends a reply

3. An MCU wants to configure a Gateway
    - The MCU sends a message to the serial port with an ID of 01 asking the device to change some configuration
    - The Gateway receives the message which has an ID the same as its ID and therefore processes the message and replies back to the MCU over the serial port. Take not that in this instance the Gateway did not send the message out over the radio because the message was addressed to the Gateway ID.

1. Modes of Operation

All of the RF modules can be configured to operate in different modes that determine the how the device will operate.

There are two modes of operation:
- Gateway Mode
- Sensor Mode

2.1 Gateway Mode

The device acts as Serial to Radio / Radio to Serial converter. Messages sent to the serial port are transmitted through the radio. Incoming messages from the Radio are sent through the Serial port. One of the main characteristics of Gateway mode is the serial port. The RF module must be connected to the serial port of the micro controller (ie. Raspberry Pi, Arduino, Beaglebone...). A Gateway is always awake , cannot be put into a low lower sleep and is usually powered by a dedicated power supply.

2.2 Sensor Mode

The device can sleep which allows it to operate for long periods of time consuming very little power (0.5 µA). The device is awoken either by an external event (e.g. door switch) or an internal timer that causes the device to wake up, send some data, and then go back to sleep. A device in sensor mode can also be attached to an external power supply and be always awake. When the device is awake it can receive and process radio messages. In this way a request/reply messaging model is created where a Gateway requests data from the sensor nodes.

2. Message Format

Each message is made up of 12 characters made up of three sections:

[a] - Message start indicator that is used to detect the start of a message
[id] - 2 charachter device ID identifying the device the message is intended for
[message] - 9 characters message content

An example of a message is: a45HELLO----
Where the device ID is 45 and the message content is “HELLO----”.

5. Message Reference

Message	**Action**	**Return message after successful transaction (*)**
+++	Use this command to determine if a device is up and running. This command can only be sent over the serial interface. If you want to test the presence of a remote device use the HELLO command. This command does not require message start indicator, or	OK-------

	device ID (refer Message Format section).	
ANAA	Analog value 0-32767 from Analog A (Flex Module Pin17)	a99ANAA99999
ANAB	Analog value 0-32767 from Analog B (Flex Module Pin18)	a99ANAB99999
ATCH[num] E.g. ATCH5	Configures the device Frequency. Only accepts a number 1 to 6 and sets the desired frequency per chart below. Device requires a restart to apply the new frequency. For two devices to communicate they must have the same frequency and channel (ATCN). Device requires a restart to apply the new channel. 1 - 433 MHZ 2 - 915 MHZ (default US & Canada) 3 - 868.3 MHZ (default Europe) 4 - 868 MHZ 5 - 903 MHZ 6 - 315 MHZ	a99ATCH9----
ATCN[num] E.g. ATCN5	Configures the transmission channel. Only accepts numbers 0-9. Each frequency (ATCH) has 10 channels. For two devices to communicate they must have the same frequency and channel. Device requires a restart to apply the new channel. Default is 0.	a99ATCN9----
ATEE[num] E.g. ATEE1	Configures whether to encrypt message. ATEE1 = encryption on ATEE0 = encryption off Default is off. Refer ATEA for the related encryption key.	a99ATEE9----
ATEA[16 chars] E.g. ATEAThisMyPrivateKey	A 16 character private key used for encryption. Wireless messages are limited to 12 characters so only 5 character passphrases can be sent over the air. 16 character passphrases can	a99ATEA[5 chars] The 5 chars are the first 5 characters of the key.**

	be set through the serial port. When encryption is turned on all devices with the same private key can communicate with each other. Default is 16 spaces.	
ATID[5 num]	A 5 digit numeric value that sets the PanID of the device. All devices of the same PanID will communicate with each other. A reboot is required to apply the new PanID. Defaults is 23205.	a99ATID99999
AWAKE (up to Version 2 - most temperature and switch sensors are on V2)	Takes the device out of sleep mode. There is a 5 second window after a device is restarted when this command can be used to wake up sleeping devices.	a99AWAKE---
WAKE (Version 3 and up - Flex modules and light sensor are on version 3 and up)	Takes the device out of sleep mode. There is a 5 second window after a device is restarted when this command can be used to wake up sleeping devices.	a99WAKE----
BATT	Transmits the input voltage. A battery message is transmitted every 10 intervals (see INTVL) or every 10 requests.	a99BATT9.99-
BME280	Transmits a temperature reading in celsius, a humidity reading % and environment pressure reading in Pascals from the external BME280 sensor.	a99TMPA99.99 a99HUM99.99- a99PA9999.9-
CHDEVID[2 numbers] E.g.: CHDEVID85	Changes the device ID to a new ID. Make sure you send two characters.	a99CHDEVID99
CYCLE	Puts the device into a cyclic sleep and is awoken to send a temperature reading every INTVL minutes. This command does not apply to Type 2&7 sensors which cannot sleep (refer TYPE).	a99CYCLE---- a--TMPA99.99 (NOMSG times) a--SLEEPING-
GPIO	Used to set GPIO pins 3,4 and 5 on the FLEX RF module high or low. GPIOAON - switch GPIO A on	a99GPIOAON--

	GPIOAOFF - switch GPIO A off GPIOBON - switch GPIO B on GPIOBOFF - switch GPIO B off GPIOCON - switch GPIO C on GPIOCOFF - switch GPIO C off	
HELLO	Replies HELLO	`a99HELLO----`
HTU21	Transmits a temperature (Celsius) and humidity (%) from the HTU21 external sensor.	`a99HUM99.99-` `a99TMPA99.99`
INTVL[3 nums] E.g. INTVL005	A 3 digit numeric value that sets the timer interval to 1-30 minutes. See Timer section for more details. Default is 0.	`a99INTVL999-`
M[8 chars] Version 5 and up	This command can be used to send custom messages to the MAX7219 display. 8 digits to be displayed on the display. Characters supported: A,B,C,D,E,F,G,H,I,J,L,N,O,P,R,S,T,U,Z a,b,c,d,e,f,g,h,i,j,l,n,o,p,r,s,t,u,v,w,x,y,z 0,1,2,3,4,5,6,7,8,9 UNDERSCORE,SQUARE BRACKETS,SPACE Decimals are coded by subtracting 13 from the ASCII number. For example “5.” = “(” or “25.75c” = “2(75c”.	`a99MHELLO---`
MX[4 chars] Version 5 and up	Provides configuration for the wireless display MAZ7219. The four character configuration is as follows: 1 : “T” = display incoming temperature and humidity readings 2&3 : the ID of the sensor transmitting the temperature and humidity readings 4 : the suffix (e.g. “c” or “F”). Can be any character	`a99MXT99F (means display temperature readings from sensor ID 99 in Fahrenheit)`
NOMSG[num] E.g. NOMSG3	A numeric value between 1 and 9 that specifies how many messages to be sent after each trigger or request.	`a99NOMSG9---`
PREAMBLE	A numeric value 1 or 0. If set to 1 then	`a99PREAMBLE9`

	the device will add additional message data to the serial port. [Index][PanID][Message][RSSI][LQI] Index - Packet Length PanID - Refer PanID section Message - 12 character message RSSI - Received Signal Strength Indicator in dBm LQI - The Link Quality Indicator estimates how easily a received signal can be demodulated	
REBOOT	Restarts the device	a99REBOOT---
RESET	Resets the device settings back to factory default. This command can only be sent over the serial port. This command does not require message start indicator, or device ID (refer Message Format section).	OK-------
SHT21	Transmits a temperature (Celsius) and humidity (%) from the SHT21 external sensor.	a99HUM99.99- a99TMPA99.99
SLEEP	Puts the device into Sleep Mode. See Sleep Mode section for more details. This command only applies to devices in sensor mode.	a99STATEON/OFF (NOMSG times) a99SLEEPING-
TEMP	Transmits a temperature reading in Celsius from the 10k thermistor	TMPA99.99--- (NOMSG times)
TEMPB	Transmits a temperature in celsius and humidity readings from DHT22	TMPB99.99---HUM99.99---- (NOMSG times)
TEMPC	Transmits a temperature reading in Celsius from the DS18B 20 sensor	TMPC99.99--- (NOMSG times)
TYPE[num] E.g. TYPE2	1=Sensor Mode (All sensors available, disables serial comms TX and RX) 2=Gateway Mode (enables serial commsn TX and RX) 3=DHT22 sleep mode 4=DS18B20 sleep mode 5=AnalogA sleep mode 6=AnalogB sleep mode 7=Enables Relay A & B and disables TX RX	a--TYPE99---

***** ‘9’ represents a number, e.g. 9.99 is a single digit number with two decimals, or 99 is a two digit number without decimals.

a99ERR------ : Returned when a command is unrecognized or the message is not correctly formatted as specified in the Message Reference.

6. Functional Reference

The RF modules are loaded with functionality typically required for building sensors and actuators connected to the internet.

All of the devices can be configured either through the serial port or over the air from one device to another. Refer the Message Processing section of this manual for more details. Normally one would configure the Gateway through the serial port, because it is connected to a micro computer (e.g. Raspberry Pi or Arduino) via the serial port, and a sensor over the air from the Gateway. Sensors come pre configured but you may want to tailor your configuration depending on your needs.

Refer to the Command Reference in the manual for all the available configurations and command syntax.

The following section describes how to configure the RF modules to your needs:

1. Gateway

Using an RF module as a Gateway connected to a Microcontroller (e.g. Raspberry Pi, Arduino, BeagleBone etc…).

In order for the device to operate as a Gateway (and enable the serial port) it must be configured as a TYPE2 sensor. The Flex is shipped as a TYPE2 and is ready to be connected to an external MCU. Take note that devices that are not TYPE2 cannot be configured through their serial ports because the serial port is only enabled once the device is set to TYPE2. In order to configure a device to TYPE 2 you need to do it over the air from another Gateway. If you do not have another Gateway then it is possible to configure any device within 5 seconds of being started up (see section on device recovery for more details).

Connect the RF Module to the MCU as follows:

Flex Pin	**Sensor Node***	**MCU Pin**	**Raspberry Pi Pin**
1	3V3	3V3	1
10	GND	GND	6
15	Tx	Rx	10
16	Rx	Tx	8

*It is possible to convert a sensor mode into a Gateway, however your will need to solder wires or pins to the board contacts labelled in the above table.

The serial port is configured at the following settings:

Baud : 9600
Data : 8
Parity : None
Stop : 1

2. Button Sensor

Using the RF Module as a door/windows sensor or to sense the opening/closing of any tactile switch.

Configurations:

- TYPE1 - configure the RF module as a TYPE1 sensor
- SLEEP - stets the sleep interval in between button state messages (refer sleep mode section for more details)
- BUTTON - returns the state of a button (BUTTONON or BUTTONOFF)
- NOMSG - sets the number of messages to be sent with each trigger
- INTVL - sets the sleep interval

Pinouts:

- For sensor node RF modules there are two contacts labelled “BtnA” (refer the board layout in Low Power Wireless Sensor Node section of this manual).
- For Flex modules use Pin 6 and Pin 10 to connect to the external switch.

Functionality:

- When the two button contacts are joined or separated (opened or closed) this action will trigger a button switch message to be transmitted.
- When the switch goes from an open state to closed then BUTTONOFF is sent.When the switch goes from a closed state to open then BUTTONON is sent.
- Opening or closing the switch will cause the device to come out of sleep mode, transmit a reading and then go back to sleep.

Battery Restrictions:

- Battery consumption is higher when the button contacts are closed (i.e. connected or closed circuit).
- Coin cell battery lifetime when button contacts are normally closed (e.g. door closed) is around 3 months. To get battery lifetime up to 12 months rather use a normally closed switch when attached to a door normally closed, or a normally open switch when connected to a door that is normally open.
- When you but the Wireless Sensor Switch we provide the option of normally open or normally closed magnetic switch.
- If purchasing your own magnetic door switches or using pre-installed switches please check the type of switch you have.

3. Thermistor Sensor

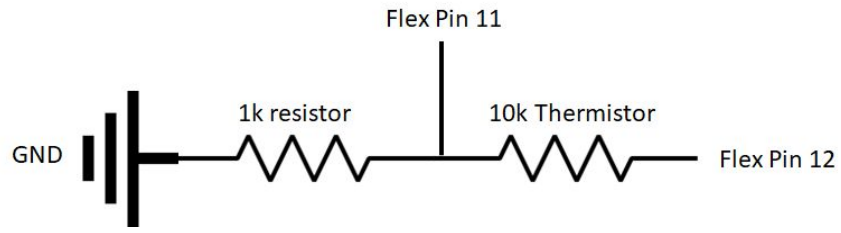
A thermistor is an analog sensor for sensing temperature. We have configured the device for an external 10K thermistor (NTCLE100E3103JB0).

Configurations:

- TYPE1 configure the RF module as a TYPE1 sensor
- CYCLE - puts the device into a cyclic sleep waking every INTVL minutes to send a temperature reading
- TEMP - sending the TEMP command to the device will cause it to transmit a temperature reading
- INTVL - sets the cycle interval
- NOMSG - sets the number of messages to be sent

Pinouts:

- For sensor node RF modules there are two contacts labelled “Tmp” (refer the board layout in Low Power Wireless Sensor Node section of this manual). A 1k resistor must also be installed on the PCB.
- For Flex module follow the following wiring diagram:



Functionality:

- A temperature reading can be requested from the module by sending it a TEMP command
- Set the device into a cyclic sleep using the INTVL and CYCLE configurations (see more details in Cycle Mode section)
- Temperature is sent in the following format ：TMPA999.99--

4. DS18B20 Temperature Sensor

DS18B20 is a digital temperature sensor. It is more accurate than and has higher sensitivity than a thermistor.

Configurations:

- TYPE4 - configure the RF module as a TYPE4 sensor
- CYCLE - puts the device into a cyclic sleep waking every INTVL minutes to send a temperature reading
- TEMP3 - sending the TEMP3 command to the device will cause it to transmit a temperature reading from the DS18B20 sensor

- INTVL - sets the cycle interval
- NOMSG - sets the number of messages to be sent

Pinouts:

- The DS18B20 pinout is not supported on the sensor node module
- For Flex module follow the following wiring table:

Flex Pin	**DS18B20 Pin**
1	3V3
8	Data
10	GND

The DS18B20 usually required a 4.7k pull up resistor between 3V3 and Data pins but it is not required with the Flex because it uses an internal pull-up resistor.

Functionality:

- A temperature reading can be requested from the module by sending it a TEMP3 command
- Set the device into a cyclic sleep using the INTVL and CYCLE configurations (see more details in Cycle Mode section)
- Temperature is sent in the following format ：TMPC99.99---
- Due to the power consumption of the DS18B20 sensor it is not recomended to use a coin cell battery to power the sensor (even in sleep mode). We calculate a coin cell battery will last for about a month sending readings every 5 minutes. Two AA batteries would be more appropriate.

5. DHT22 Temperature and Humidity Sensor

DHT22 is a digital temperature and humidity sensor. It is more accurate and has higher sensitivity than a thermistor.

Configurations:

- TYPE3 - configure the RF module as a TYPE3 sensor
- CYCLE - puts the device into a cyclic sleep waking every INTVL minutes to send a temperature reading
- TEMP2 - sending the TEMP2 command to the device will cause it to transmit a temperature reading from the DHT22 sensor
- INTVL - sets the cycle interval
- NOMSG - sets the number of messages to be sent

Pinouts:

- The DHT22 pinout is not supported on the sensor node module
- For Flex module follow the following wiring table:

Flex Pin	DHT22 Pin
External Power Source	3.3V - 6V
7	Data
10	GND

The DHT22 usually required a 10k pull up resistor between 3V3 and Data pins but it is not required with the Flex because it uses an internal pull-up resistor.

Functionality:

- A temperature and humidity reading can be requested from the module by sending it a TEMP2 command
- Set the device into a cyclic sleep using the INTVL and CYCLE configurations (see more details in Cycle Mode section)
- Temperature is sent in the following format ：TMPC99.99---
- Humidity is sent in the following format ：HUM99.99---
- Due to the power consumption of the DS18B20 sensor it is not recomended to use a coin cell battery to power the sensor (even in sleep mode). We calculate a coin cell battery will last for about a month sending readings every 5 minutes. Two AA batteries would be more appropriate.

6. Analog Sensor

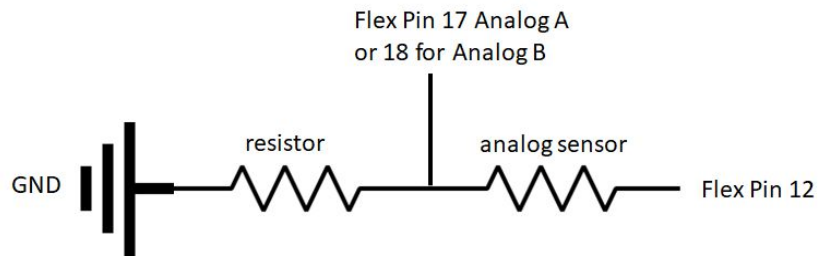
The Flex module has two analog input pins that will return values between 0 and 32767.

Configurations:

- TYPE5 - configure the RF module as a TYPE5 for analog sensor 5.
- TYPE6 - configure the RF module as a TYPE6 for analog sensor 6.
- CYCLE - puts the device into a cyclic sleep waking every INTVL minutes to send a temperature reading
- ANAA - sending the ANAA command to the device will cause it to transmit the analog value for pin analog sensor A
- ANAB - sending the ANAB command to the device will cause it to transmit the analog value for pin analog sensor B
- INTVL - sets the cycle interval
- NOMSG - sets the number of messages to be sent

Pinouts:

- The analog sensor pinouts are not supported on the sensor node module
- For Flex module follow the following wiring diagram:



The size of the resistor depends on the analog sensor being used. Input voltage from Pin 12 is 3.3V.

Functionality:

- An analog reading can be requested from the module by sending it a ANAA or ANAB command
- Set the device into a cyclic sleep using the INTVL and CYCLE configurations (see more details in Cycle Mode section). Only one analog device reading (either from sensor A or B depending on the TYPE you set) can be sent when in a sleep cycle.
- The analog reading is a number between 0 and 32,767 and is sent in the following format : ANAA99999--- or ANAB99999---

7. Relay switch

The sensor node and Flex module have two relay control pins that can be used to switch an external relay switch.

Configurations:

- TYPE7 - configure the RF module as a TYPE7 for to use the relays switches.
- CYCLE - Not applicable
- SLEEP - Not applicable
- RELAYAON - switch relay A on
- RELAYAOFF - switch relay A off
- RELAYBON - switch relay B on
- RELAYBOFF - switch relay B off
- NOMSG - sets the number of messages to be sent

Pinouts:

Sensor Node	**Flex**	**External Relay**
Tx	15	Relay A IN
Rx	16	Relay B IN
GND	10	GND
Do not connect	Do not connect	External power

The sensor node does support relay switching but wires or pins will need to be soldered to the Tx and Rx through holes on the sensor node board.

Functionality:

- The RF module must be always awake for it to receive messages to switch the relay
- Send the device RELAYAON, RELAYBON, RELAYAOFF and RELAYBOFF messages to switch RelayA and.or RelayB on/off.

8. GPIO

The Flex module has three GPIO pins configured for output that can be used to switch each GPIO on/off. When on they supply 3.3V and when off they supply 0V. Useful for controlling LED's.

Configurations:

- TYPE1 - configure the RF module as a TYPE1 for to use the GPIO outputs.
- CYCLE - Not applicable
- SLEEP - Not applicable
- GPIOA1 - switch GPIO A on
- GPIOA0 - switch GPIO A off
- GPIOB1 - switch GPIO B on
- GPIOB0 - switch GPIO B off
- GPIOC1 - switch GPIO C on
- GPIOC0 - switch GPIO C off
- NOMSG - Not applicable

Pinouts:

Flex	**External Device/Circuit**
5	+3.3V
4	+3.3V
3	+3.3V
10	GND

The sensor node does not support GPIO switching.

Functionality:

- Send the device GPIOA1, GPIOB1, GPIOC1, GPIOA0, GPIOB0 and GPIOC0 messages to switch GPIO A,B or C high/low.

9. BME280 Temperature, Humidity & Pressure Sensor

Data from this sensor is transmitted over 5 messages, 40 bytes in total and shown in the table below:

	Start Char	Sensor ID		Data								
Message 1->	a	9	9	B	M	P	-	-	1	2	3	4
Message 2->	a	9	9	5	6	7	8	9	10	11	12	13
Message 3->	a	9	9	14	15	16	17	18	19	20	21	22
Message 4->	a	9	9	23	24	25	26	27	28	29	30	31
Message 5->	a	9	9	32	33	34	35	36	37	38	39	40

“BMP--” is the marker for the start of a BME280 sensor reading. The 40 bytes, indicated in red in the above diagram, represent the data elements from the sensor. In this manual we will not go into the conversion of these data elements to the actual temperature, humidity and pressure readings but these details can be found in bme280.py, or in the Bosch BME280 datasheet.

When the number of messages (NOMSG) setting is set to more than 1 then duplicates need to be removed before decoding the 40 bytes. Refer rflib.py for an example on how to remove duplicates. It is recommended to use a NOMSG setting of 2 or 3 to improve transmission reliability.

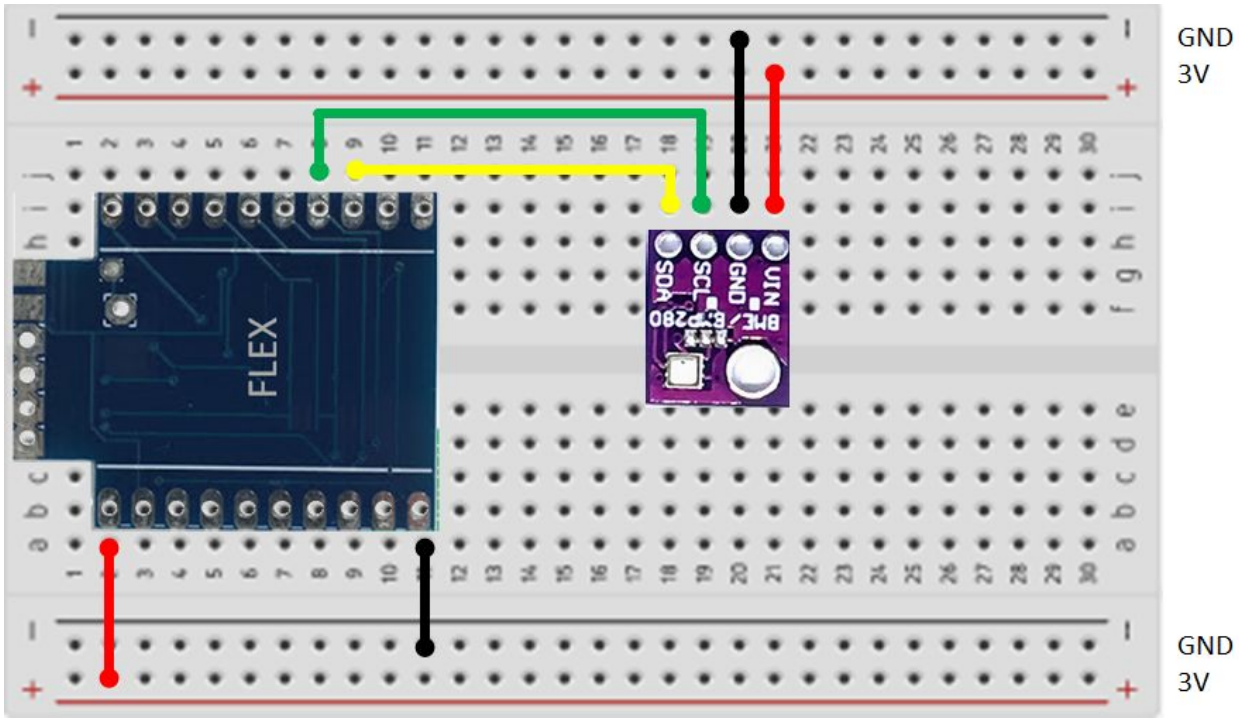
We have incorporated bme280.py into our utilities (serial_mon.py and rf_config.py) so that BME280 sensor data is converted to a standard human readable radio message format shown below:

```
Temperature reading   : a99TMPA99.99 (Celsius)
Humidity reading           : a99HUM99.99- (%)
Pressure reading           : a99PA9999.9- (Pascals)
```

Configurations:

- TYPE9 - configure the RF module as a TYPE9 for BME280
- CYCLE - puts the device into a cyclic sleep waking every INTVL minutes to send the sensor readings
- INTVL - sets the cycle interval
- NOMSG - sets the number of messages to be sent. If set to 2 then 10 messages sent, 3 then 15 messages sent etc.. The order in which they are sent is message 1 x NOMSG, message2 x NOMSG, message3 x NOMSG etc..
- BMP280 - Requests a set readings from the sensor

Flex RF module wiring diagram for BME280 sensor



You may be wondering why we decided to transmit raw sensor data over the air instead of doing the conversion on the radio device before transmission. The reason for this is due to the computational overheads (ie memory) required to perform the conversion is beyond what we can afford in the RF module.

10. Personal Area Networks

All messages sent over the Radio and received by sensors of the same PanID. The sensor PanID can be configured - see ATID command in the Command Reference. Therefore using the PanID sensors can be grouped together to form Message Networks. A device's unique identifier is it's PanID + Device ID.

When changing the PanID from a Gateway to a sensor node make sure you change the PanID of the sensor nodes before changing the PanID of the Gateway.

11. Connecting to a PC COM port using a USB cable

An RF module can be connected to a PC using an FTDI cable.

Pin 2 FTDI Rx - Connect to Rx on the RF module
Pin 3 FTDI Tx - Connect to Tx on the RF module

Pin 4 FTDI 3v3 - Connect to 3v3 on the RF module
Pin 6 FTDI GND - Connect to GND on the RF module

Caution!!! Make sure that the FTDI is set to 3V3 operation. Many FTDI converters are 5V or have a 5V/3V3 dip switch to change voltage. A 5V FTDI can damage the RF module!!!

11. Startup

When a device is powered on, or after it reboots (refer REBOOT command), it enters into a startup sequence which begins with sending STARTED-- 5 times. If the device has been configured for Sleep or Cycle modes (refer Sleep Mode section of this manual) the device will stay awake for 5 seconds after each restart allowing you time to cancel sleep mode if you wish. The device will also send STARTED-- to the serial port on startup. Allow the device to complete the 5 second startup sequence before interacting with the device.

12. Recovering a dead device

There may be instances where a device has been misconfigured and you want to reset it back to factory defaults. This can be done by issuing the device the RESET command. This command can only be sent through the serial port and cannot be sent over the air to sensor nodes. To reset sensor nodes you will need to solder wires or use small alligator clips to connect to the serial port (Tx,Rx,3.3V and GND).

The RESET command must be sent during the 5 second startup sequence. The RSET command does not follow the a99AAAAAAAAA message format. Only send “RESET” to the serial port of the device during startup and then the device will reset back to factory settings.

13. Sleep Timer

The sleep timer can be used to wake the device from sleep every INTVL minutes. The timer can be used to either send sensor readings every INTVL minutes or send the button state (STATEON/STATEOFF) every INTVL minutes. See Sleep Mode section of this manual for more details.

14. Sleep Mode

Sleep mode causes a device to operate using very little (0.5ua) power consumption when asleep. The device can run on battery power for long periods of time (up to a year on a coin cell battery depending on transmission intervals).

Use the CYCLE command to put a device to sleep in a cyclic temperature sleep mode. The device will awaken every INTVL minutes and send a temperature reading and then go back to sleep.

The device can be put to sleep using the SLEEP command. If the device was put to sleep using the SLEEP command and INTVL is non zero then the device will awaken every INTVL minutes and transmit the button state (STATEON-- or STATEOFF-) and then go back to sleep.

Regardless of how the device was put to sleep (CYCLE or SLEEP) it will awaken if the button switch is triggered and it will transmit BUTTONON- or BUTTONOFF. This feature allows you to use a sensor node as a temperature sens and a button sensor at the same time.

When a device is asleep then it cannot accept incoming messages from the radio.

The device will transmit SLEEPING immediately before going to sleep.

15. Waking up a device

An RF module can be awakened by powering it off/on and then sending it the WAKE (*A*WAKE in version 2 and below) command within 5 seconds. The WAKE command will bring the device out of sleep mode.

Note: A sensor will only last 20 minutes on a coin cell battery when not asleep.

The device will transmit WAKE when a device awakens.