BitPipe™ shield for Arduino

User Guide

Version 1.1

31/10/2017

Table of Contents

1. Revision History

Revision	Date	Comments
1.0	13/04/2017	First release
1.1	31/10/2017	Update section 6.1 (importing the BitPipe™ library)

2. Introduction

The BitPipe™ Arduino shield is designed to provide connectivity between the Briowireless BitPipe™ and an Arduino board. It is designed for developers, integrators and hobbyists who want to easily integrate cellular Internet connectivity to their hardware solutions.

This document describes how to setup and use the BitPipe™ shield for Arduino.

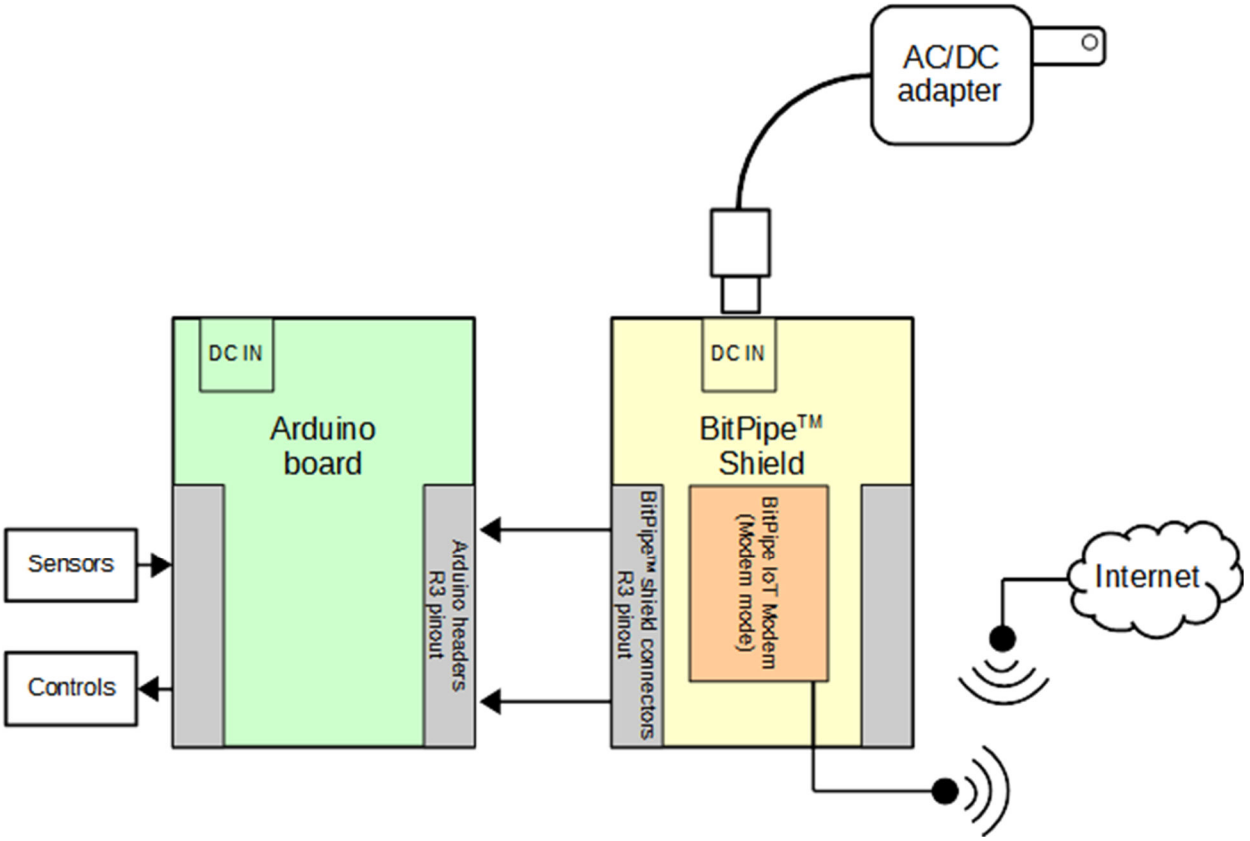


Figure 1: BitPipe™ shield Simplified Block Diagram

2.1. Audience

This document targets software integrators that are experienced with the Arduino environment.

This document supposes the use of the Arduino IDE for development.

3. Hardware Set-up

3.1. Target Arduino Hardware

The BitPipe™ shield is designed to work with the Arduino R3 pinout. The following Arduino boards are compatible with the BitPipe™ shield:

- Arduino Uno
- Arduino Leonardo
- Arduino Mega & Mega ADK
- Arduino Zero
- Arduino Due
- Arduino M0 & M0 Pro

Other Arduino boards can also be used, although additional steps and verifications may be required by the user. Refer to section 5 for alternative boards instructions.

3.2. What's included

The following items are part of the solution:

- 1 x BitPipe™ shield
- 1 x Phillips M2 type screw (for fastening BitPipe™ to the shield)
- 1 x 6 position female header (J11)
- 2 x 8 position female header (J5 and J12)
- 1 x 10 position female header (J6)
- 5 x header jumpers

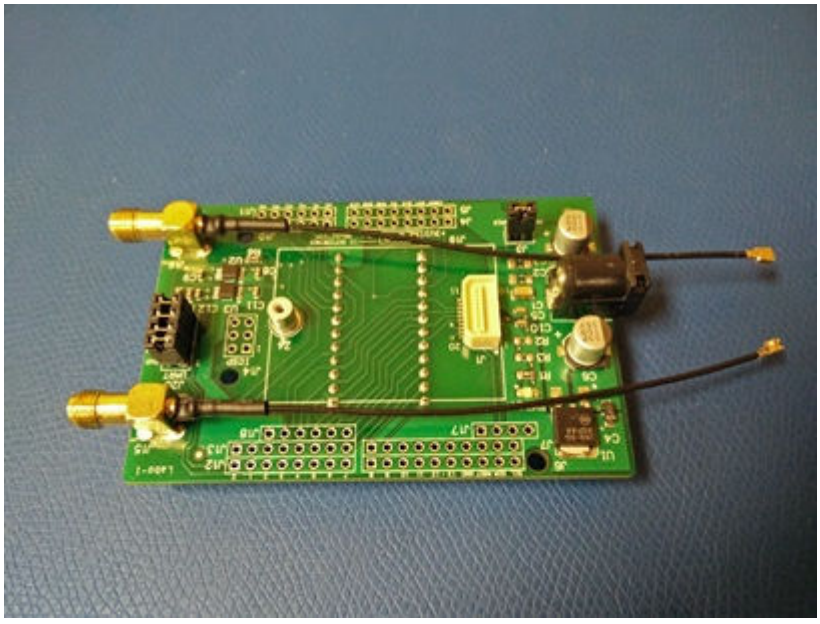
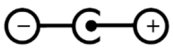


Figure 2: BitPipe™ Arduino shield

3.3. Things you will need

The following items are required to properly set-up a BitPipe™ shield to an Arduino board:

1. A suggested Arduino board (See section 3.1);
2. A BitPipe™ (go to www.briowireless.com for available product models);
3. A SIM card (3FF), activated with a data plan;
4. 9VDC power adaptor (refer to BitPipe™ manual for maximum current consumption), provides power to the Arduino board and BitPipe™.
    - IMPORTANT: Connector must be of positive polarity: 
    - Mating barrel power connector (2.1mm I.D., 5.5mm O.D.)
5. Antennas (visit www.briowireless.com to buy them);
6. If you are planning to use a small form factor Arduino (Mini/Micro/Nano) and do not want to solder it to the BitPipe™ shield, you will need two 12 positions 100mils pitch female headers (not included).

3.4. Electro-mechanical assembly

The following steps describe how to assemble the BitPipe™ shield, BitPipe™ and Arduino board:

> **IMPORTANT:** *Industry standard electronic device and component handling methods should be utilized to protect the hardware from damage that may be caused from electro-static discharge (ESD) or fast-transient bursts (FTB). It is recommended to utilize a properly grounded wrist strap when handling these devices to ensure their long-term reliability.*

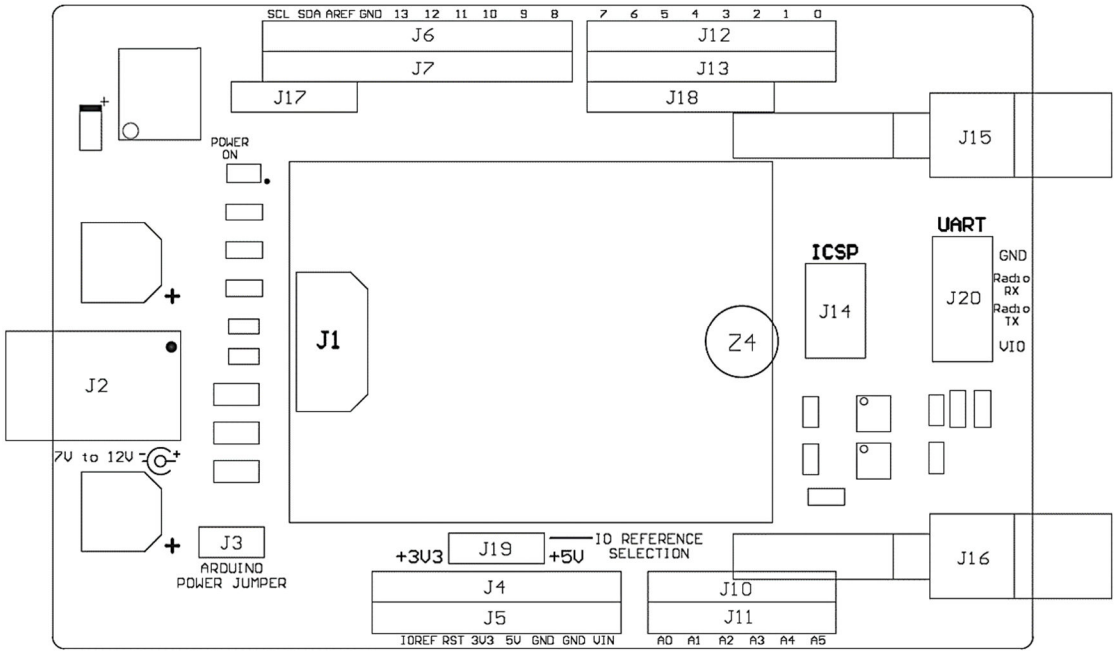


Figure 3: BitPipe™ shield connectors

1. The first step is to solder the Arduino shield headers on the shield itself.

If using a regular form factor Arduino board, solder the included female headers to connectors J5, J6, J11 and J12 (refer to Figure 3) on the top side of the shield (Figure 4).

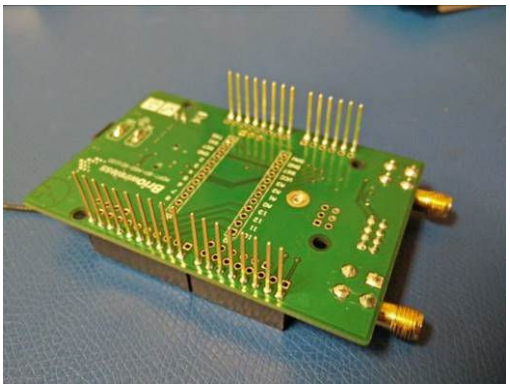


Figure 4: Arduino R3 headers

2. Install four jumpers on J20. To power the Arduino board from the BitPipe™ shield, install one jumper on J3 as well.

3. Carefully align and insert the BitPipe™ shield connectors to the target Arduino board. Do not use excessive force as this may damage the connector(s). See Figure 5 and Figure 6.

Figure 5: Arduino M0 Pro installed

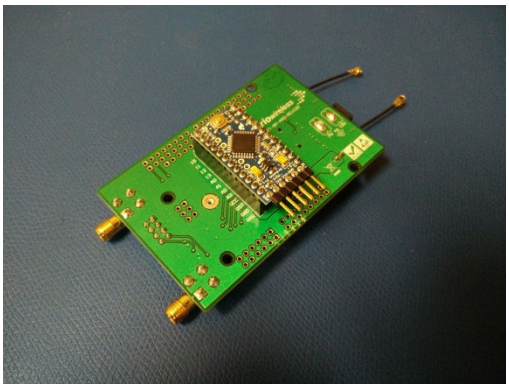


Figure 6: Arduino Pro Mini installed

4. Insert and push the SIM card into the SIM cardholder located at the front of the BitPipe™. A “click” should be felt when it is fully inserted. The SIM cardholder is a Push-in / Push-out type. See Figure 7 and Figure 8.

Figure 7: SIM installation

Figure 8: SIM installation

5. Carefully align and install the BitPipe™ to the shield BitPipe™ connector. Do not use excessive force as this may damage the connector(s). See Figure 9 and Figure 10.

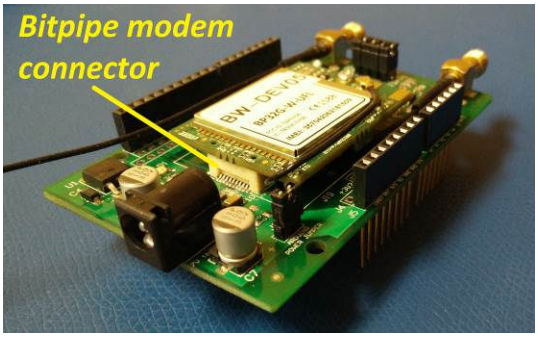


Figure 9: BitPipe™ modem connector

6. Fasten the BitPipe™ to the shield using the M2 screw.

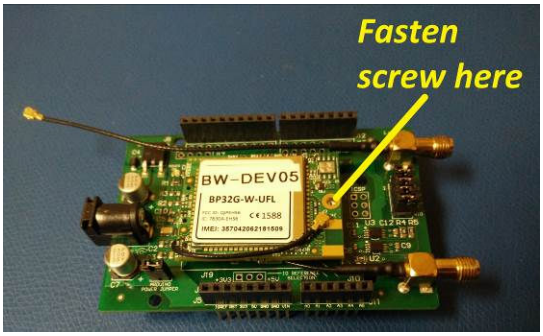


Figure 10: BitPipe™ fastening screw

7. Connect the shield antenna(s) to the BitPipe™ UFL connector. Carefully line up the connector and apply light pressure downwards until you feel a 'click' to confirm it is properly inserted. Do not force it down as this may damage the connector or BitPipe™ device.

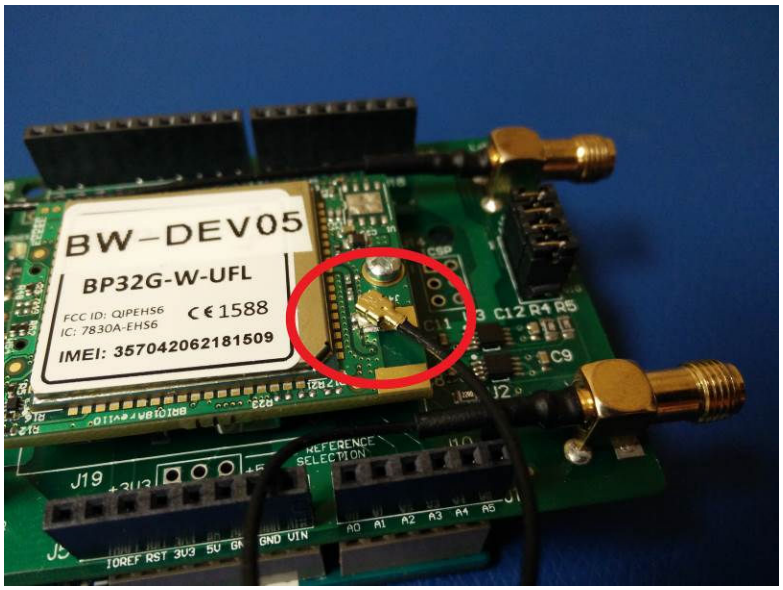


Figure 11: UFL antenna connector

3.5. Power & communication

1. By default, the BitPipe™ serial port is attached to Arduino serial port (0 & 1) through jumper J20 pins 3-4 and 5-6. This connection is the primary method of communicating commands and data from the Arduino board to the BitPipe™.

2. If you wish to power the Arduino board from the same supply as the BitPipe™ shield, closing jumper J3 will deliver power to the Arduino's RAW input. When powering the Arduino board from a different supply, **including a USB cable**, jumper J3 must be open.

3. Insert power supply 9V DC connector into the BitPipe™ shield and plug the power adapter into a suitable AC mains receptacle. A green LED (D1) should turn on nearby the top left corner of the shield, indicating power is available to the board.

4. Hardware Interconnect

The following describes the interconnect details and signals used to communicate and control a BitPipe™ via the serial interface.

4.1. BitPipe™ shield header connectors signal assignments

The BitPipe™ shield connectors are used to provide radio control to the Arduino board.

- BitPipe™ shield pin 0 is mapped to UART-TX on the BitPipe™ and connected to the Arduino pin 0 (RX). BitPipe™ UART-TX can be disconnected from the Arduino pin 0 (RX) by removing jumper J20 5-6 to be able to update your sketch through the Arduino bootloader or use another serial port if you have more than one port, like the Arduino Mega.
- BitPipe™ shield pin 1 is mapped to UART-RX on the BitPipe™ and connected to the Arduino pin 1 (TX). BitPipe™ UART-RX can be disconnected from the Arduino pin 1 (TX) by removing jumper J20 3-4 to be able to update your sketch through the Arduino bootloader or use another serial port if you have more than one port, like the Arduino Mega.
- BitPipe™ shield IOREF is used as IO voltage reference for serial communication and connected through pins 7-8 of J20. If your Arduino board does not have the R3 pinout, this pin is not available. In that case, you will need to place a jumper between pins 1-2 of J19 if your Arduino board operates at 3.3V or pins 2-3 of J19 if your Arduino board operates at 5V.

Arduino connector		BitPipe™ shield header		Comments
Pin(s)	**Signal**	**Pin(s)**	**Signal**	
RAW	Raw input supply	VIN	DC In supply	Power input from barrel connector
GND	Ground	GND	Ground	Common ground
IOREF	IO reference	IOREF	IO Reference	IO voltage reference of the Arduino
0	Serial RX	0	BitPipe™ UART-TX	Arduino RX/BitPipe™ TX
1	Serial TX	1	BitPipe™ UART-RX	Arduino TX/BitPipe™ RX

Table 1: BitPipe™ shield header to Arduino header interconnect

NOTE: All other BitPipe™ shield pins are not used.

4.2. BitPipe™ Modem mode

The BitPipe™ has 2 different operating modes: Modem mode and Autonomous mode. Hardware strappings on the BitPipe™ shield set the BitPipe™ in Modem mode by default. In this mode, the BitPipe™ Serial API is enabled.

For more information on Serial API, visit https://briowireless.com/products/Module-API-SDK-Documentation/index.html.

4.3. Controlling radio power

Radio power is controlled via the Serial API.

See examples source to know how to control radio power.

5. Alternative Arduino boards

5.1. Old Arduino boards (Pre R3 Headers)

Some Arduino Boards may have the old Arduino headers that do not include a pin dedicated for IOREF. On these boards, the user will be required to provide the IOREF in order for the UART to function properly. This can be achieved in two different ways.

The first method is to populate the header J19 with a 3 position 100 mils male header and to install a jumper to select the correct voltage for your Arduino board. This method gives the user the flexibility to modify their voltage selection with ease. Refer to Figure 12 for an example of correct jumper placement.

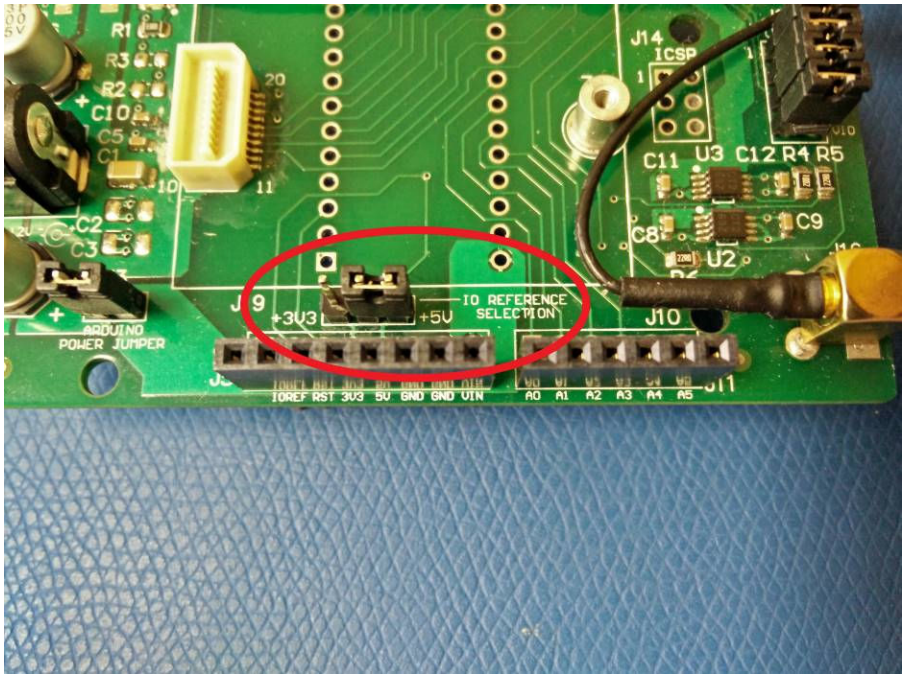


Figure 12: Jumper example for 5V IOREF selection

The second method is to simply create a solder bridge or to solder a wire between the two desired positions on the J19 connector.

Note that the required IOREF voltage depends on the Arduino board being used – please ensure to select the correct voltage for your application. Using the wrong voltage could result in permanent damage to the Arduino board.

5.2. Small form factor Arduino boards

If using a small form factor Arduino board, you can either solder 12 position 100 mils pitch female headers (for removable installation) on the bottom side of the shield (Figure 13), or solder the Arduino board itself on the shield.

Arduino Mini and Nano are compatible with the BitPipe™ headers. Note that with the Arduino Nano, some pins will exceed the headers and that pin numbers may be skewed. Arduino Micro can also be used, but must be offset on the shield connectors so that the power and communication pins are matching.

In any case, the user must validate pinout compatibility between the Arduino board and the BitPipe™ shield.

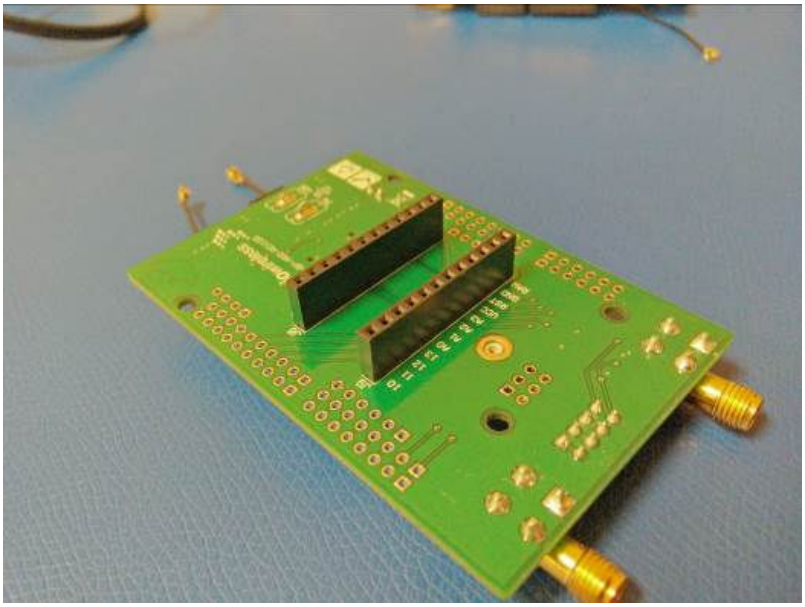


Figure 13 – Arduino shield with small form factor headers

6. Arduino environment

6.1. Importing BitPipe™ Arduino SDK into Arduino IDE

The BitPipe™ SDK has been integrated in an Arduino library suitable to be imported like any other third-party library into Arduino IDE.

To import the BitPipe™ SDK into Arduino IDE:

1. Download the latest BitPipe™ Arduino libraries on your PC from https://briowireless.com/support/ in the Arduino section.
2. Unzip the BitPipe_Arduino folder you just download it. The folder "src" contains the following libraries:
    a. bitpipe_api: library that enables communication with the bitpipe module.
    b. message_protocol: utility used by the bitpipe_api to send and receive messages
    c. MQTTSimplified: library used by the example MQTTSimplified.
3. Add the 3 libraries to Arduino IDE. Repeat steps (a) and (b) for each library.
    a. In Arduino IDE, select Sketch->Include Library->Add .ZIP Library…
    b. Choose a subfolder in "src" and click Open.
4. You will find examples in File->Examples->BitPipe.

6.2. Examples

There are 4 examples to demonstrate how to use the BitPipe™ SDK.

Before running any example, you must change definitions of `SIM_PIN`, `SIM_PUK`, `APN_CONFIG_HOST`, `APN_CONFIG_USER`, `APN_CONFIG_PASS` to the settings of your SIM and APN.

Arduino LED behavior (connected on pin 13):

1. Small flash every second: Arduino is running normally.
2. Solid green: Unable to initialize the library. Check your `bp_api_init_params` structure for any invalid value.
3. Slow flash (1Hz): Example completed with success.
4. Fast flash (2.5Hz): No SIM card is inserted.
5. Very fast flash (10Hz): Error status returned from the library.

6.2.1. HTTPGet

This example sends a GET request to httpbin.org and waits for a reply. It demonstrates how the HTTP API can be used to fetch data from a web server. Nothing is done with the data returned.

6.2.2. HTTPPost

This example sends a POST request with data to httpbin.org. It demonstrates how the HTTP API can be used to post data to a web server. Nothing is done with the data returned.

6.2.3. TCPClient

This example sends data to a TCP server and is expecting an echo. It demonstrates how the TCP API can be used to connect and exchange data with a TCP server. Nothing is done with the data returned.

6.2.4. MQTT

This example demonstrates how to use the MQTT API to develop your IoT device. It connects to an MQTT broker, sends simulated temperature and humidity in topic `sensors/temperature` and `sensors/humidity`, and listens to topic `outputs/relay<x>` (digital outputs) and `outputs/pwm<y>` (pwm outputs). `X` can be any value between 0 and 3 and `Y` can be any value between 0 and 1. You can increase or decrease the number of outputs by changing `MAX_RELAY_OUTPUTS` and `MAX_PWM_OUTPUTS`. Output pins numbers have been choosen to work with the Arduino 4 Relays shield. All values sent or received are transferred as plain text.

Before running the example, adjust `MQTT_REMOTE_HOST`, `MQTT_REMOTE_PORT`, `MQTT_CLIENT_ID`, `MQTT_USER` and `MQTT_PASS` to your needs.

6.2.5. SMS

This example sends a random string to the configured phone number by the text messaging API. The replied data is read, but not used in the current example.

6.3. Creating your own project

You can start from any example and modify it to suit your needs. The important parts are:

1. Initialize the library by calling `bp_api_init` before trying to send or receive data to/from the BitPipe™. Create a `bp_api_init_params` structure and initialize it like this:
    a. `app_callbacks`: A `bp_api_callbacks` structure pointer that will hold function pointers used like callbacks by your application. Non-used callbacks must be set to NULL.
    b. `uart_write`: A function that write data to the UART. The function has to block until all bytes are sent.
    c. `deserialize_buf`/`serialize_buf`: Buffers used to encode/decode packets exchanged between the Arduino and the BitPipe™. Can be any size between 48 and 260, but must be large enough to hold the biggest message you send to the BitPipe™.
2. Call `bp_api_process_incoming` with received data. You can call this function for every byte you receive or one time with many bytes. When a packet is fully received, the callback suitable for the message received will be called before returning from `bp_api_process_incoming`.
3. The status returned by the API is the first indication if you have a problem or not.
4. In your callback, always verify the status received, it indicates if your request is accepted or not by the BitPipe™.

7. Support

For additional support, please refer to the following resources that may help you during your evaluation and development using the Briowireless BitPipe™ and related products.

Because the BitPipe™ Arduino library is based on the BitPipe™ SDK, refer to the Module API SDK documentation for more information about the library.

Email: support@briowireless.com

Online: For online support and downloads please go to:
http://briowireless.com/support/