AdafruitBluefruitSPI Library Documentation

Release 1.0

Kevin Townsend

Jan 20, 2020

Contents

Helper class to work with the Adafruit Bluefruit LE SPI Friend.

CHAPTER 1

Dependencies

This driver depends on:

- Adafruit CircuitPython
- Bus Device

Please ensure all dependencies are available on the CircuitPython filesystem. This is easily achieved by downloading the Adafruit library and driver bundle.

CHAPTER 2

Installing from PyPI

On supported GNU/Linux systems like the Raspberry Pi, you can install the driver locally from PyPI. To install for current user:

```
pip3 install adafruit-circuitpython-bluefruitspi
```

To install system-wide (this may be required in some cases):

```
sudo pip3 install adafruit-circuitpython-bluefruitspi
```

To install in a virtual environment in your current project:

```
mkdir project-name && cd project-name
python3 -m venv .env
source .env/bin/activate
pip3 install adafruit-circuitpython-bluefruitspi
```

CHAPTER 3

Usage Example

```
# A simple echo test for the Feather M0 Bluefruit
# Sets the name, then echo's all RX'd data with a reversed packet

import time
import busio
import board
from digitalio import DigitalInOut
from adafruit_bluefruitspi import BluefruitSPI

spi_bus = busio.SPI(board.SCK, MOSI=board.MOSI, MISO=board.MISO)
cs = DigitalInOut(board.D8)
irq = DigitalInOut(board.D7)
rst = DigitalInOut(board.D4)
bluefruit = BluefruitSPI(spi_bus, cs, irq, rst, debug=False)

# Initialize the device and perform a factory reset
print("Initializing the Bluefruit LE SPI Friend module")
bluefruit.init()
bluefruit.command_check_OK(b'AT+FACTORYRESET', delay=1)

# Print the response to 'ATI' (info request) as a string
print(str(bluefruit.command_check_OK(b'ATI'), 'utf-8'))

# Change advertised name
bluefruit.command_check_OK(b'AT+GAPDEVNAME=BlinkaBLE')

while True:
    print("Waiting for a connection to Bluefruit LE Connect ...")
    # Wait for a connection ...
    dotcount = 0
    while not bluefruit.connected:
        print(".", end="")
        dotcount = (dotcount + 1) % 80
        if dotcount == 79:
```

(continues on next page)

(continued from previous page)

```
            print("")
        time.sleep(0.5)

    # Once connected, check for incoming BLE UART data
    print("\n *Connected!*")
    connection_timestamp = time.monotonic()
    while True:
        # Check our connection status every 3 seconds
        if time.monotonic() - connection_timestamp > 3:
            connection_timestamp = time.monotonic()
            if not bluefruit.connected:
                break

        # OK we're still connected, see if we have any data waiting
        resp = bluefruit.uart_rx()
        if not resp:
            continue  # nothin'
        print("Read %d bytes: %s" % (len(resp), resp))
        # Now write it!
        print("Writing reverse...")
        send = []
        for i in range(len(resp), 0, -1):
            send.append(resp[i-1])
        print(bytes(send))
        bluefruit.uart_tx(bytes(send))

    print("Connection lost.")
```

CHAPTER 4

Contributing

Contributions are welcome! Please read our Code of Conduct before contributing to help this project stay welcoming.

CHAPTER 5

Documentation

For information on building library documentation, please check out this guide.

CHAPTER 6

Table of Contents

6.1 Simple test

Ensure your device works with this simple test.

Listing 1: examples/bluefruitspi_simpletest.py

```
# A simple echo test for the Feather M0 Bluefruit
# Sets the name, then echo's all RX'd data with a reversed packet

import time
import busio
import board
from digitalio import DigitalInOut
from adafruit_bluefruitspi import BluefruitSPI

spi_bus = busio.SPI(board.SCK, MOSI=board.MOSI, MISO=board.MISO)
cs = DigitalInOut(board.D8)
irq = DigitalInOut(board.D7)
rst = DigitalInOut(board.D4)
bluefruit = BluefruitSPI(spi_bus, cs, irq, rst, debug=False)

# Initialize the device and perform a factory reset
print("Initializing the Bluefruit LE SPI Friend module")
bluefruit.init()
bluefruit.command_check_OK(b'AT+FACTORYRESET', delay=1)

# Print the response to 'ATI' (info request) as a string
print(str(bluefruit.command_check_OK(b'ATI'), 'utf-8'))

# Change advertised name
bluefruit.command_check_OK(b'AT+GAPDEVNAME=BlinkaBLE')

while True:
```

(continues on next page)

(continued from previous page)

```
    print("Waiting for a connection to Bluefruit LE Connect ...")
    # Wait for a connection ...
    dotcount = 0
    while not bluefruit.connected:
        print(".", end="")
        dotcount = (dotcount + 1) % 80
        if dotcount == 79:
            print("")
        time.sleep(0.5)

    # Once connected, check for incoming BLE UART data
    print("\n *Connected!*")
    connection_timestamp = time.monotonic()
    while True:
        # Check our connection status every 3 seconds
        if time.monotonic() - connection_timestamp > 3:
            connection_timestamp = time.monotonic()
            if not bluefruit.connected:
                break

        # OK we're still connected, see if we have any data waiting
        resp = bluefruit.uart_rx()
        if not resp:
            continue  # nothin'
        print("Read %d bytes: %s" % (len(resp), resp))
        # Now write it!
        print("Writing reverse...")
        send = []
        for i in range(len(resp), 0, -1):
            send.append(resp[i-1])
        print(bytes(send))
        bluefruit.uart_tx(bytes(send))

    print("Connection lost.")
```

6.2 adafruit_bluefruitspi

Helper class to work with the Adafruit Bluefruit LE SPI friend breakout.

- Author(s): Kevin Townsend

6.2.1 Implementation Notes

Hardware:

"* Adafruit Bluefruit LE SPI Friend"

Software and Dependencies:

- Adafruit CircuitPython firmware for the supported boards: https://github.com/adafruit/circuitpython/releases
- Adafruit's Bus Device library: https://github.com/adafruit/Adafruit_CircuitPython_BusDevice

class adafruit_bluefruitspi.**BluefruitSPI**(*spi*, *cs*, *irq*, *reset*, *debug=False*)
Helper for the Bluefruit LE SPI Friend

command(*string*)
Send a command and check response code

command_check_OK(*command*, *delay=0.0*)
Send a fully formed bytestring AT command, and check whether we got an 'OK' back. Returns payload bytes if there is any

connected
Whether the Bluefruit module is connected to the central

init()
Sends the SDEP initialize command, which causes the board to reset. This command should complete in under 1s.

read_packet()
Will read a Bluefruit Connect packet and return it in a parsed format. Currently supports Button and Color packets only

uart_rx()
Reads byte data from the BLE UART FIFO.

uart_tx(*data*)
Sends the specific bytestring out over BLE UART. :param data: The bytestring to send.

CHAPTER 7

Indices and tables

- genindex
- modindex
- search

Python Module Index

a

Index