Serial (UART) communication and Global Positioning System

Written By: Eli Foster

This document was generated on 2024-08-25 10:18:56 AM (MST).

INTRODUCTION

UART is another form of communication between devices that, while outdated, can still be useful for many components. This guide will introduce you to UART concepts and show how they can be used with components. Learn more about the technical side of UART here.

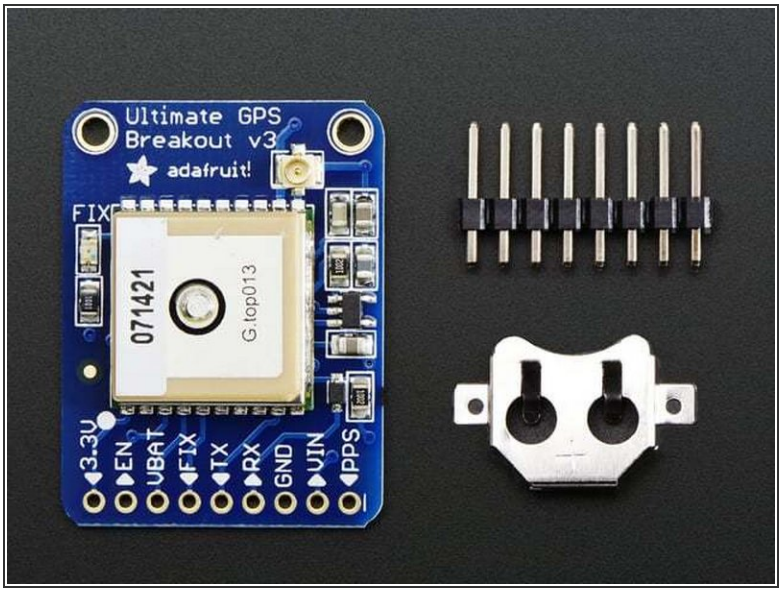
TOOLS:

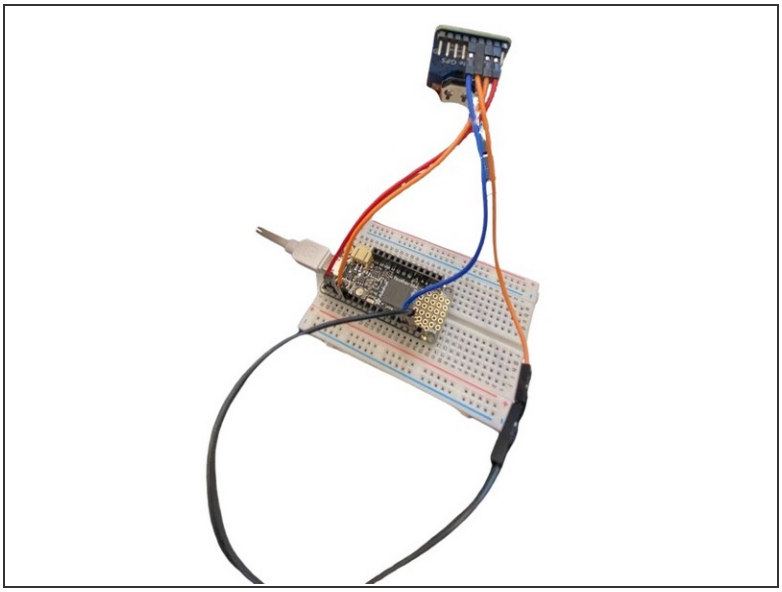
- Micro-USB Cable (1)
- Breadboard Jumper cables (4)
  F-M

PARTS:

- Feather M4 Express (1)
- Ultimate GPS Breakout V3 (1)

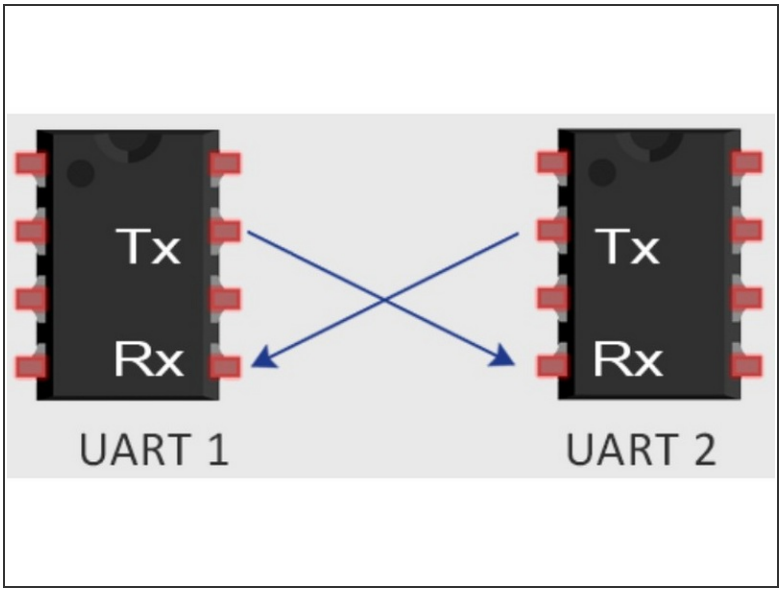
Step 1 — Prerequisites

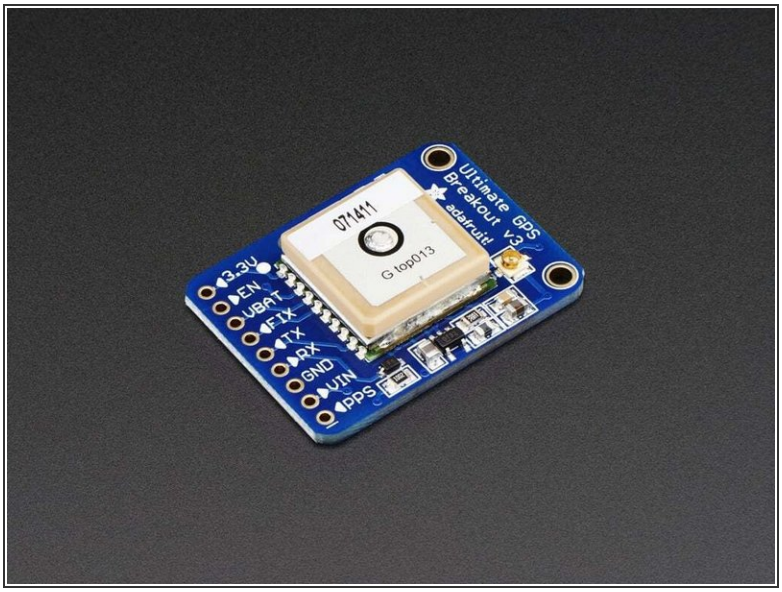




- Before you complete this tutorial, make sure that you have completed the Feather M4 Express tutorial first.
- You will also need the Ultimate GPS Breakout V3 from Adafruit and 4 female-to-male jumper cables.

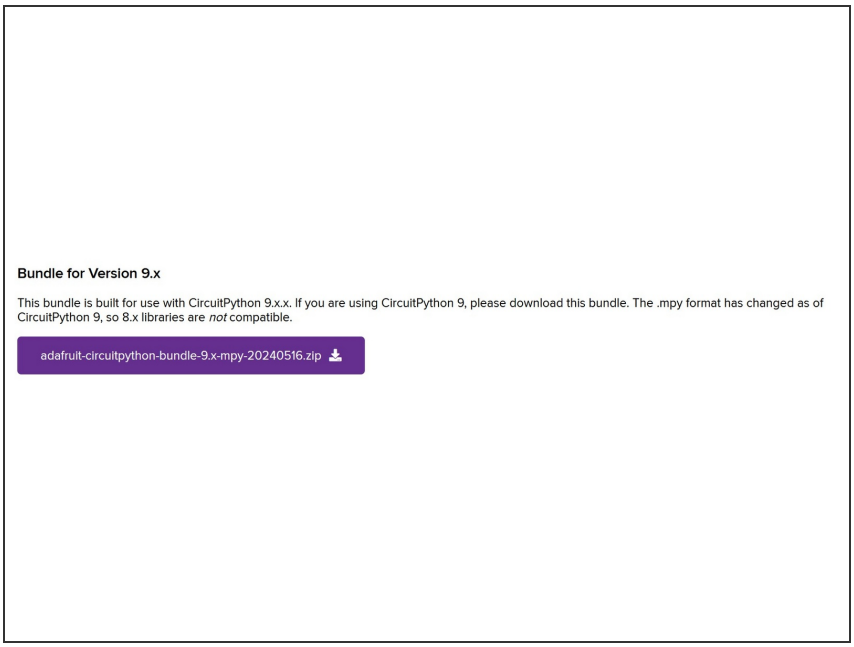
Step 2 — Hardware





- UART is a serial method of communication. This means that it is synchronous transmission between two devices, one singular bit at a time. There is no parallel communication. This makes it significantly slower than most modern protocols (based on the baud rate), but some boards still only use UART(due to its simplicity), so it's important to learn.
- UART works by simply connecting a "transmit" pin to a "receive" pin, or TX and RX. Usually, these two are intertwined, each device having an RX and TX pin. Bridging these with jumper cables allows communication from each respective TX pin to the other device's RX pin. To learn more visit this link.
- Start with the blue GPS breakout board. In this case, the Feather will be UART 1 and the breakout board will be UART 2. To connect this specific breakout board to the Feather, connect VIN to 3V on the Feather and GND to ground. Then take jumper cables on TX and RX and attach them to the opposite label on the Feather. That's it! All connected.

Step 3 — Setup

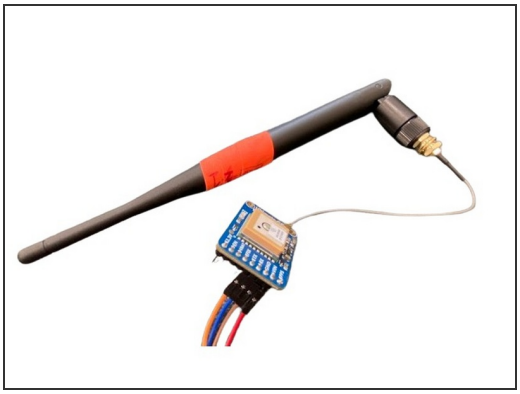


- To use CircuitPython for this tutorial, you will need the **adafruit_gps.mpy** library from the common bundle.
- *Make sure the version matches your installed CircuitPython version*.

Step 4 — Code

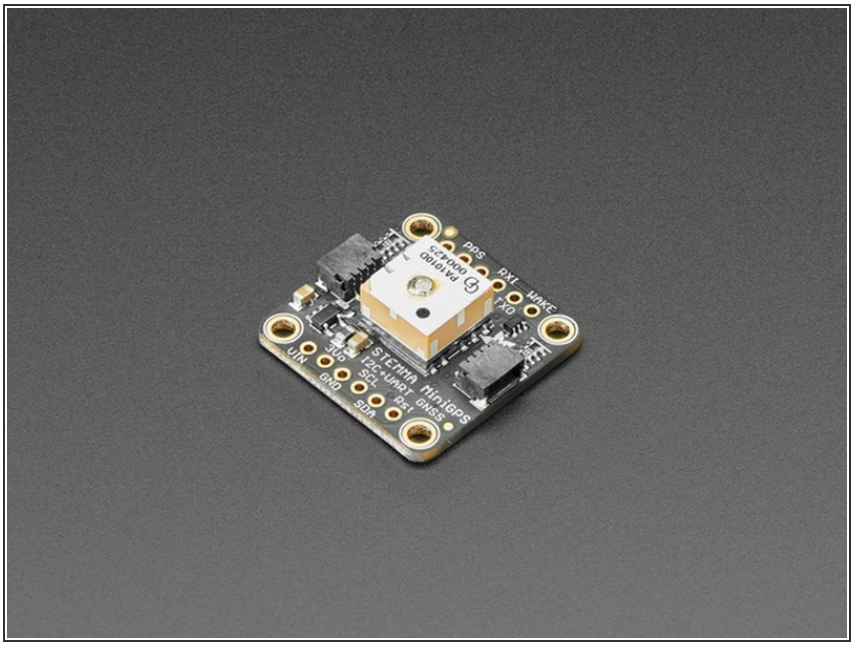
```
    print("Fix quality: {}".format(gps.fix_quality))
    # Some attributes beyond latitude, longitude and timestamp are optional
    # and might not be present.  Check if they're None before trying to use
    if gps.satellites is not None:
        print("# satellites: {}".format(gps.satellites))
    if gps.altitude_m is not None:
        print("Altitude: {} meters".format(gps.altitude_m))
    if gps.speed_knots is not None:
        print("Speed: {} knots".format(gps.speed_knots))
    if gps.track_angle_deg is not None:
        print("Track angle: {} degrees".format(gps.track_angle_deg))
    if gps.horizontal_dilution is not None:
        print("Horizontal dilution: {}".format(gps.horizontal_dilution))
    if gps.height_geoid is not None:
        print("Height geoid: {} meters".format(gps.height_geoid))
```

```
=======================================
Fix timestamp: 6/4/2024 16:27:16
Latitude: 40.955093 degrees
Longitude: -76.881195 degrees
Precise Latitude: 40 degs, 57.3064 mins
Precise Longitude: -76 degs, 52.8730 mins
Fix quality: 1
# satellites: 5
Altitude: 191.1 meters
Speed: 0.04 knots
Track angle: 128.73 degrees
Horizontal dilution: 1.78
Height geoid: -33.5 meters
```



- After defining the GPS object with UART or I2C, there are some common commands that are useful for the board. **gps.update()** updates the GPS status and re-requests any and all satellites it can reach for data. **gps.satellites**, **gps.altitude_m**, **gps.speed_knots**, **gps.track_angle_deg** are all relatively self-explanatory.
- **gps.has_fix** is a boolean that will tell you if the GPS was able to be reached and has information to give. It's important to check for this because this board only has an integrated antenna, which means it is quite inconsistent in receiving data. Checking for this variable in a loop allows your code to keep running instead of simply erring.
- Use the Adafruit example code to make sure that your board is hooked up correctly. If you only receive *Waiting for a fix...*, you may need to move the antenna on the board. If having trouble, use the exterior antenna included in the kit to extend range. This may enable you to get a signal from indoors. If not, you may need to go out back to a clear area.

Step 5 — Challenge



- To earn a badge for UART and GPS, use the other GPS breakout in the kit to connect to UART. This board uses the same libraries (and maybe even same code). Use the Adafruit website to find its pinout and get it working with UART.

Other uses for GPS include vehicle systems, location and speed tracking, and other common GPS outputs. UART can be used with an incredible number of components, all which will work the same as you used it today.