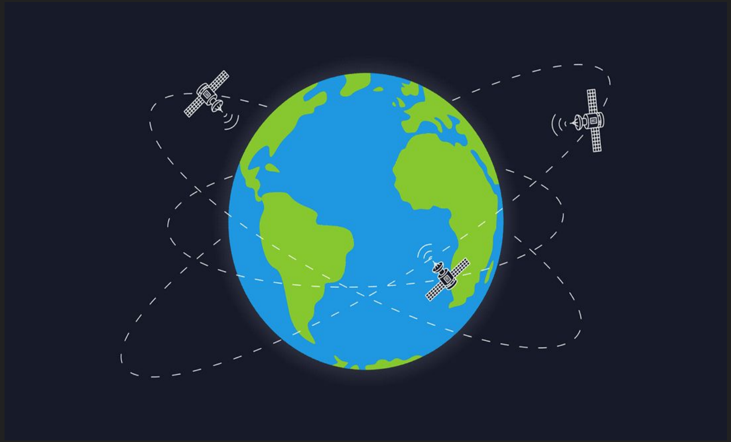
Ultimate GPS

Jeremy, Pavan, Ayush, Sam

What is GPS?

- Global Positioning System (GPS) is a satellite-based radionavigation system.
- Provides geolocation and time information to a GPS receiver from satellites.
- Receivers are programmed to receive information about where each satellite is located.



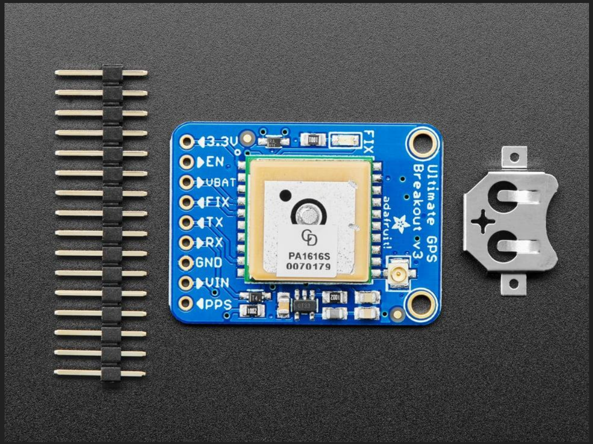
Ultimate GPS breakout module

It is built around the MTK3339 chipset which:

- can track up to 22 satellites on 66 channels
- has an excellent high-sensitivity receiver (-165 dBm tracking)
- has a built-in antenna

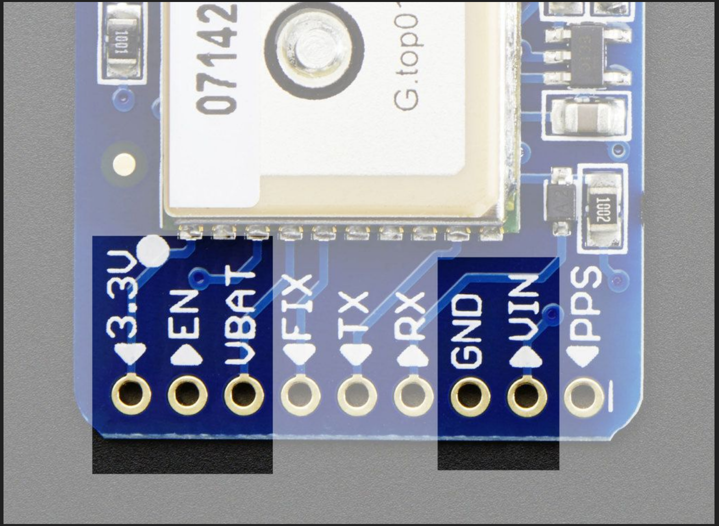
Other features:

- Built-in datalogging (uses the microcontroller and flash memory)
- an ultra-low dropout 3.3V regulator
- ENABLE pin (to turn off module using microcontroller pin or switch)
- CR1220 coin cell to keep the RTC running
- a tiny bright red LED



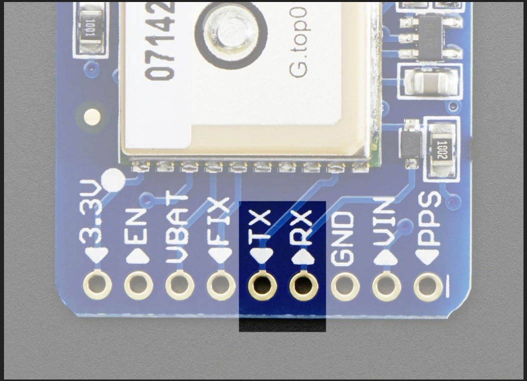
Breakout Power Pins

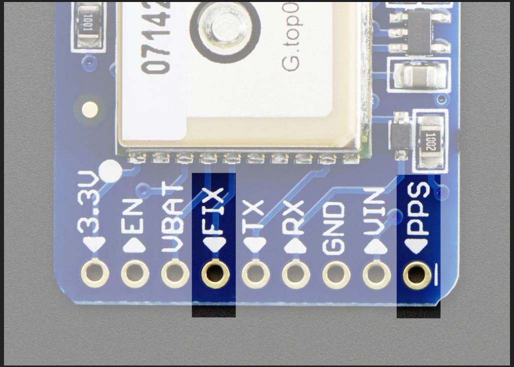
- **VIN** - power input, 3-5 VDC
- **GND** - power and signal ground
- **VBAT** - GPS RTC battery backup
- **EN** - Enable pin
- **3.3V** - output from onboard 3.3V regulator

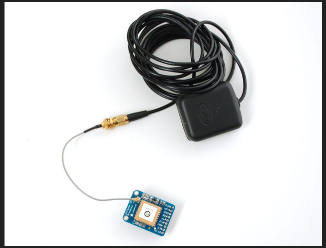


Breakout Serial Data and Other Pins

- **TX** - transmits data *from* GPS module to microcontroller
  - 3.3V logic level, 9600 baud default
- **RX** - sends data *to* GPS
  - 3.3V or 5V logic, 9600 baud default
- **FIX** - output pin and drives red LED
  - No Fix: pulses up and down once a second
  - Fix: pulses every 15 seconds for 200 ms
- **PPS** - “pulse per second” output for syncing
- Support for optional external antennas!

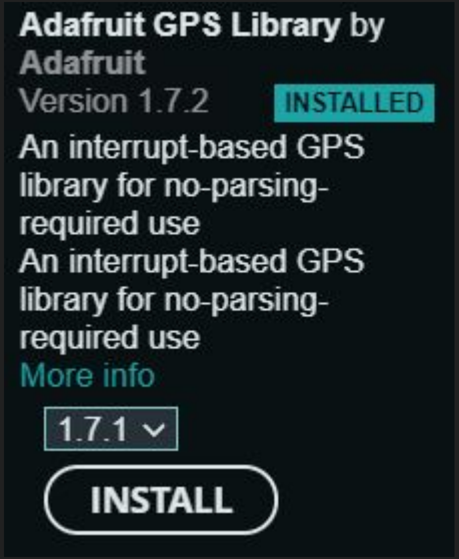






Code (Setup and Initialization)

Download the Library



Adafruit GPS Library by **Adafruit**

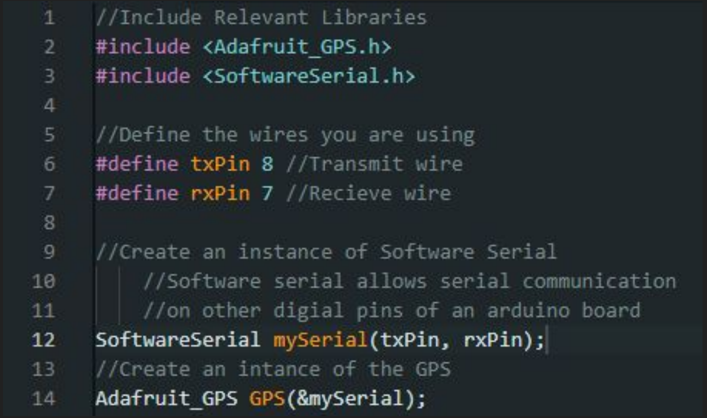
Version 1.7.2 INSTALLED

An interrupt-based GPS library for no-parsing-required use

An interrupt-based GPS library for no-parsing-required use

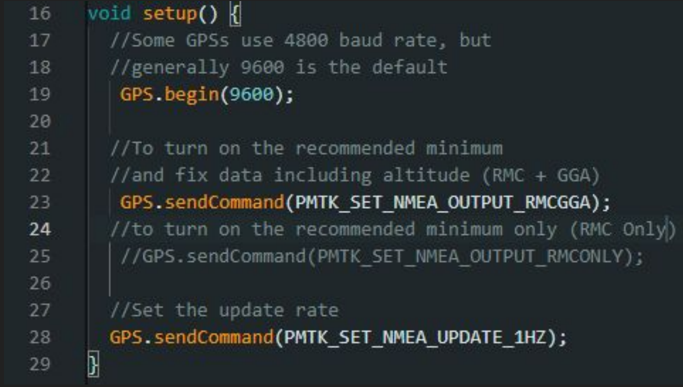
More info

Include Libraries and Define Ports



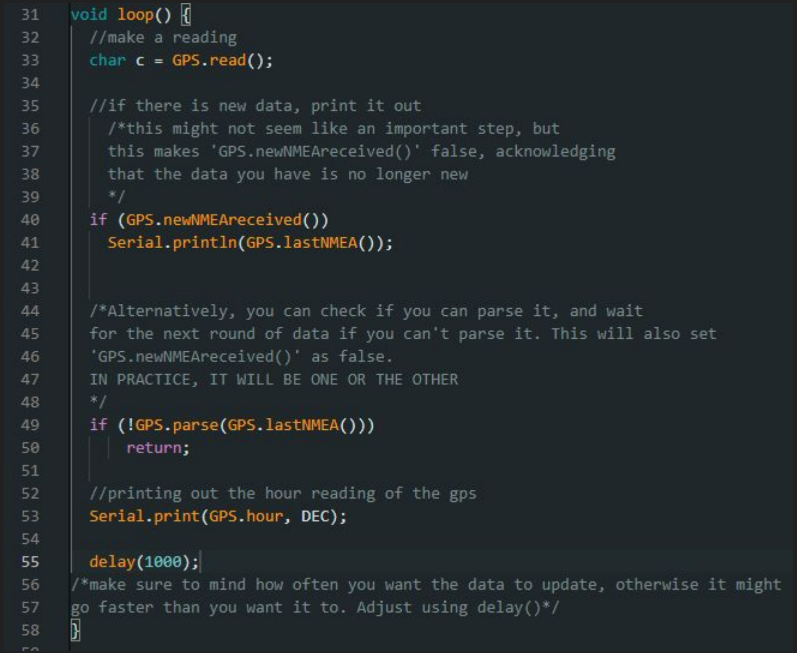
```
//Include Relevant Libraries
#include <Adafruit_GPS.h>
#include <SoftwareSerial.h>

//Define the wires you are using
#define txPin 8 //Transmit wire
#define rxPin 7 //Recieve wire

//Create an instance of Software Serial
    //Software serial allows serial communication
    //on other digial pins of an arduino board
SoftwareSerial mySerial(txPin, rxPin);
//Create an intance of the GPS
Adafruit_GPS GPS(&mySerial);
```

void setup()



```
void setup() {
  //Some GPSs use 4800 baud rate, but
  //generally 9600 is the default
   GPS.begin(9600);

  //To turn on the recommended minimum
  //and fix data including altitude (RMC + GGA)
   GPS.sendCommand(PMTK_SET_NMEA_OUTPUT_RMCGGA);
  //to turn on the recommended minimum only (RMC Only)
   //GPS.sendCommand(PMTK_SET_NMEA_OUTPUT_RMCONLY);

  //Set the update rate
  GPS.sendCommand(PMTK_SET_NMEA_UPDATE_1HZ);
}
```

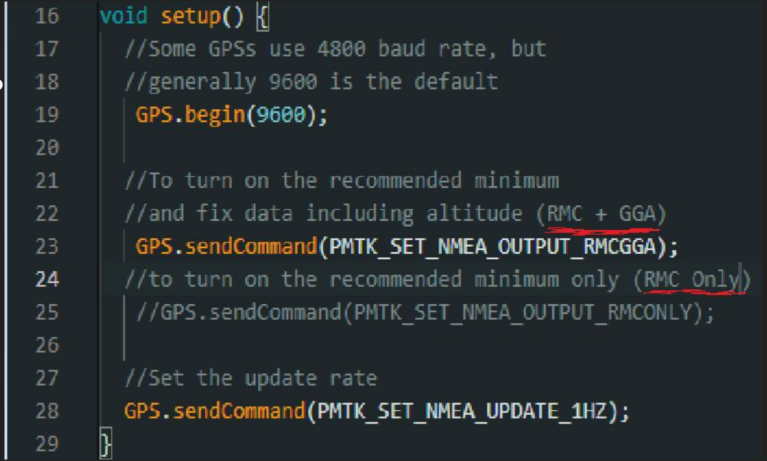
void loop()



```
void loop() {
  //make a reading
  char c = GPS.read();

  //if there is new data, print it out
    /*this might not seem like an important step, but
    this makes 'GPS.newNMEAreceived()' false, acknowledging
    that the data you have is no longer new
    */
  if (GPS.newNMEAreceived())
    Serial.println(GPS.lastNMEA());


  /*Alternatively, you can check if you can parse it, and wait
  for the next round of data if you can't parse it. This will also set
  'GPS.newNMEAreceived()' as false.
  IN PRACTICE, IT WILL BE ONE OR THE OTHER
  */
  if (!GPS.parse(GPS.lastNMEA()))
      return;

  //printing out the hour reading of the gps
  Serial.print(GPS.hour, DEC);

  delay(1000);
/*make sure to mind how often you want the data to update, otherwise it might
go faster than you want it to. Adjust using delay()*/
}
```

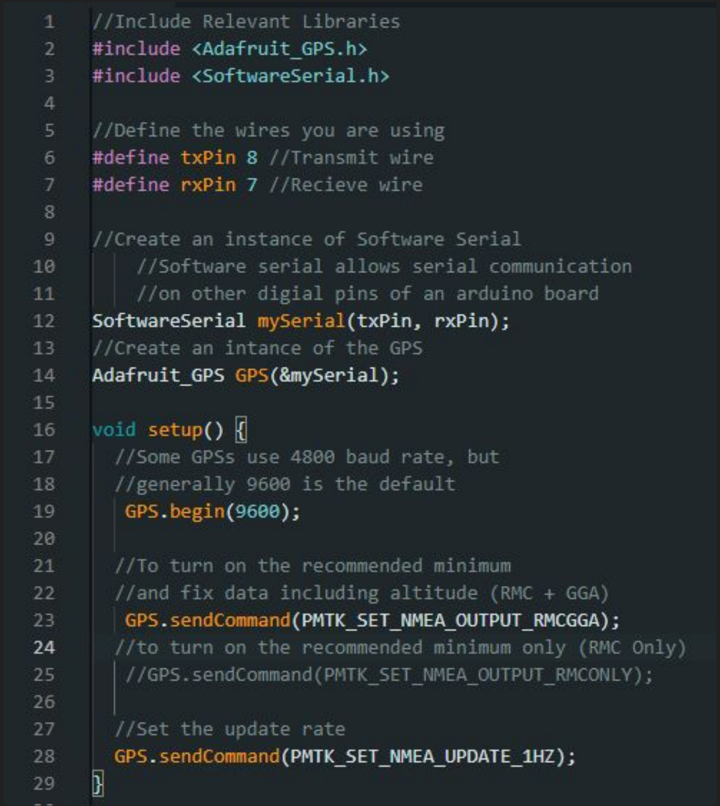
Data	Function	Data	Function
Millisecond	GPS.milliseconds	Latitude	GPS.lat
Year	GPS.year	Longitude	GPS.lon
Angle	GPS.angle	Latitude in degrees	GPS.latitudeDegrees
Satellites(GGA)	GPS.satellites	Longitude in degrees	GPS.longitudeDegrees
Altitude(GGA)	GPS.altitude	Speed (knots)	GPS.speed
Fix (RMC)	GPS.fix	Quality (GGA)	GPS.quality

Remember in setup?

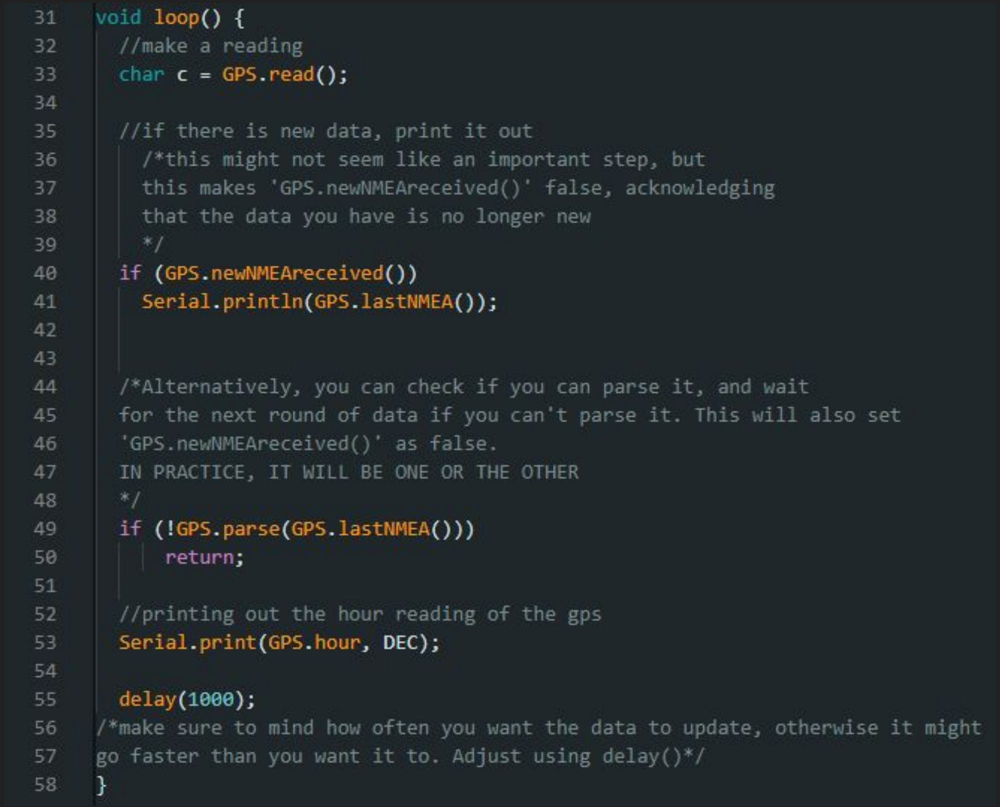


```
void setup() {
  //Some GPSs use 4800 baud rate, but
  //generally 9600 is the default
   GPS.begin(9600);

  //To turn on the recommended minimum
  //and fix data including altitude (RMC + GGA)
   GPS.sendCommand(PMTK_SET_NMEA_OUTPUT_RMCGGA);
  //to turn on the recommended minimum only (RMC Only)
   //GPS.sendCommand(PMTK_SET_NMEA_OUTPUT_RMCONLY);

  //Set the update rate
  GPS.sendCommand(PMTK_SET_NMEA_UPDATE_1HZ);
}
```

Code Summary

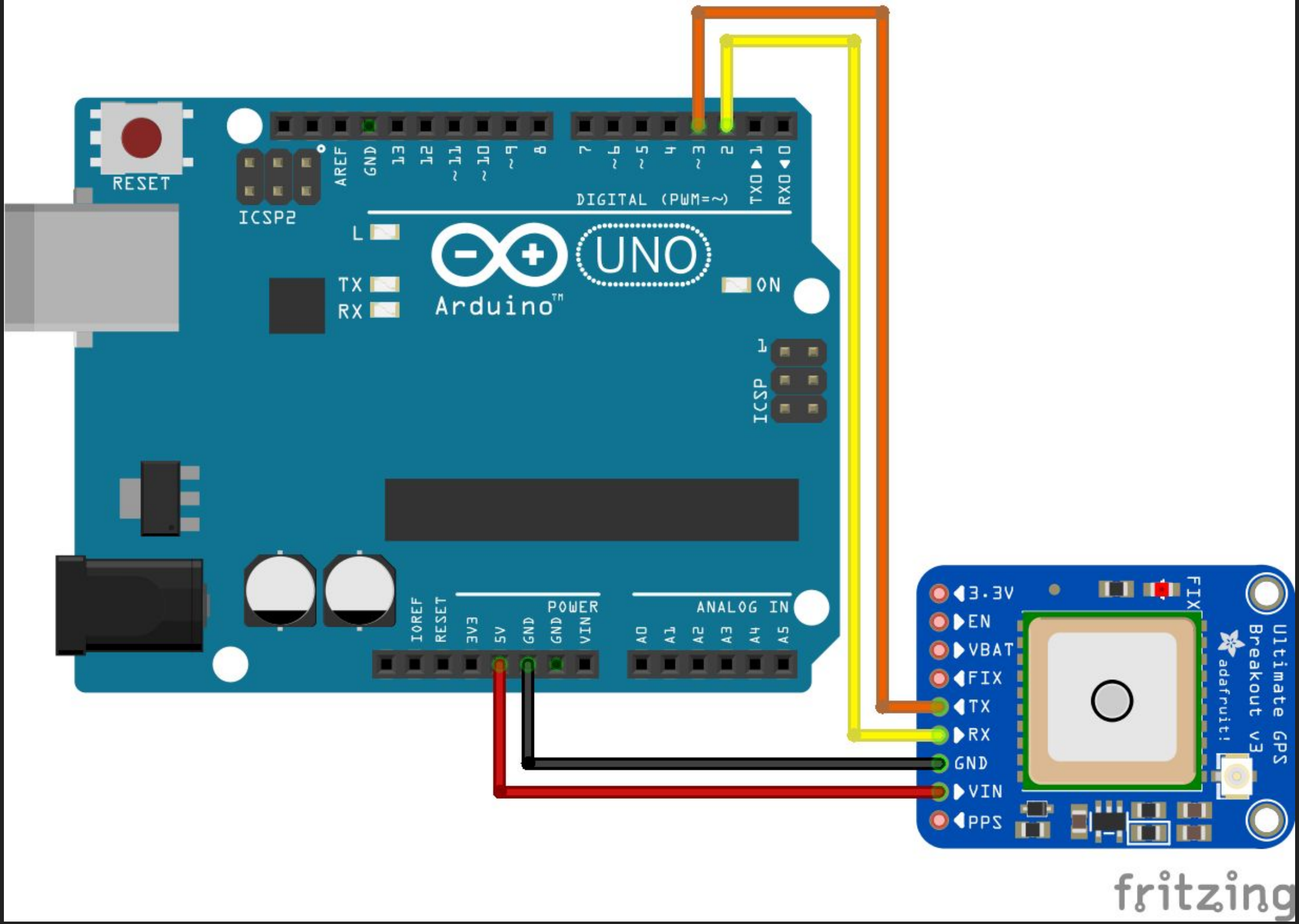


```
//Include Relevant Libraries
#include <Adafruit_GPS.h>
#include <SoftwareSerial.h>

//Define the wires you are using
#define txPin 8 //Transmit wire
#define rxPin 7 //Recieve wire

//Create an instance of Software Serial
    //Software serial allows serial communication
    //on other digial pins of an arduino board
SoftwareSerial mySerial(txPin, rxPin);
//Create an intance of the GPS
Adafruit_GPS GPS(&mySerial);

void setup() {
  //Some GPSs use 4800 baud rate, but
  //generally 9600 is the default
   GPS.begin(9600);

  //To turn on the recommended minimum
  //and fix data including altitude (RMC + GGA)
   GPS.sendCommand(PMTK_SET_NMEA_OUTPUT_RMCGGA);
  //to turn on the recommended minimum only (RMC Only)
   //GPS.sendCommand(PMTK_SET_NMEA_OUTPUT_RMCONLY);

  //Set the update rate
  GPS.sendCommand(PMTK_SET_NMEA_UPDATE_1HZ);
}
```



```
void loop() {
  //make a reading
  char c = GPS.read();

  //if there is new data, print it out
    /*this might not seem like an important step, but
    this makes 'GPS.newNMEAreceived()' false, acknowledging
    that the data you have is no longer new
    */
  if (GPS.newNMEAreceived())
    Serial.println(GPS.lastNMEA());


  /*Alternatively, you can check if you can parse it, and wait
  for the next round of data if you can't parse it. This will also set
  'GPS.newNMEAreceived()' as false.
  IN PRACTICE, IT WILL BE ONE OR THE OTHER
  */
  if (!GPS.parse(GPS.lastNMEA()))
      return;

  //printing out the hour reading of the gps
  Serial.print(GPS.hour, DEC);

  delay(1000);
/*make sure to mind how often you want the data to update, otherwise it might
go faster than you want it to. Adjust using delay()*/
}
```

Circuit


RESET
ICSP2
AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX▶1
RX◀0
DIGITAL (PWM=~)
L
TX
RX
UNO
Arduino™
ON
1
ICSP
IOREF
RESET
3V3
5V
GND
GND
VIN
POWER
ANALOG IN
A0
A1
A2
A3
A4
A5
3.3V
EN
VBAT
FIX
TX
RX
GND
VIN
PPS
FIX
Ultimate GPS Breakout v3
adafruit!
fritzing

Work Cited/Links

https://docs.arduino.cc/learn/built-in-libraries/software-serial

https://www.elithecomputerguy.com/2020/06/arduino-gps-adafruit-ultimate-gps-coordinates-and-speed-display-on-lcd/

https://learn.adafruit.com/adafruit-ultimate-gps/arduino-wiring?view=all

https://education.nationalgeographic.org/resource/gps

https://www.adafruit.com/product/746

https://core-electronics.com.au/guides/how-to-use-gps-with-arduino/

Thank you!