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

1.7.1  INSTALL

## Include Libraries and Define Ports

```cpp
#include <Adafruit_GPS.h>
#include <SoftwareSerial.h>

#define txPin 8  //Transmit wire
#define rxPin 7  //Receive wire

SoftwareSerial mySerial(txPin, rxPin);
Adafruit_GPS GPS(&mySerial);
```

## void setup()

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

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

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

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()

// 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.hours, 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()*/
```c
#include <Adafruit_GPS.h>
#include <SoftwareSerial.h>

#define txPin 8 //Transmit wire
#define rxPin 7 //Receive wire

//Create an instance of Software Serial
//Software serial allows serial communication
//on other digital pins of an arduino board
SoftwareSerial mySerial(txPin, rxPin);

//Create an instance of the GPS
Adafruit_GPS GPS(&mySerial);

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.newNMEAread()' false, acknowledging
     that the data you have is no longer new */
  if (GPS.newNMEAread())
    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.newNMEAread()' 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
Work Cited/Links

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


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

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

Thank you!