LaACES
Student
Ballooning
Course

The Ultimate GPS Logger Shield – GPS

LaACES Student Ballooning Course

LaACES Student Ballooning Course

Introduction

- Now that you've learned about the Global Positioning System, it's time to learn about the GPS receiver you'll be using - the Adafruit Ultimate GPS Logger Shield

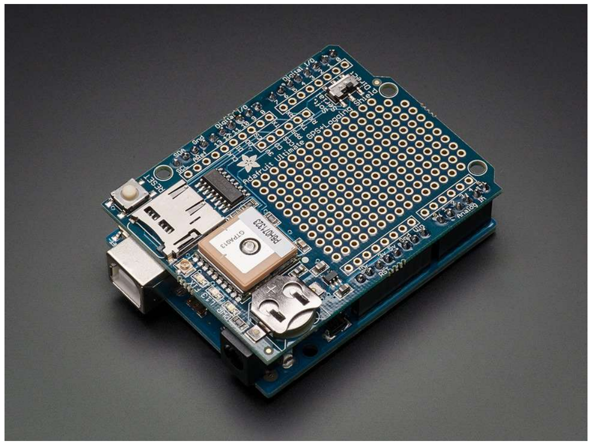


Figure 1: Shown is the Adafruit Ultimate GPS Logger Shield. It is stacked on an Arduino Uno

LaACES Student Ballooning Course

Adafruit Ultimate GPS Logger Shield Overview

- This is the Arduino Ultimate GPS Logger Shield. It comes with
  - microSD socket
  - Internal antenna and connector for external antenna
  - GPS unit
  - RTC and coin cell holder
  - Prototyping Area

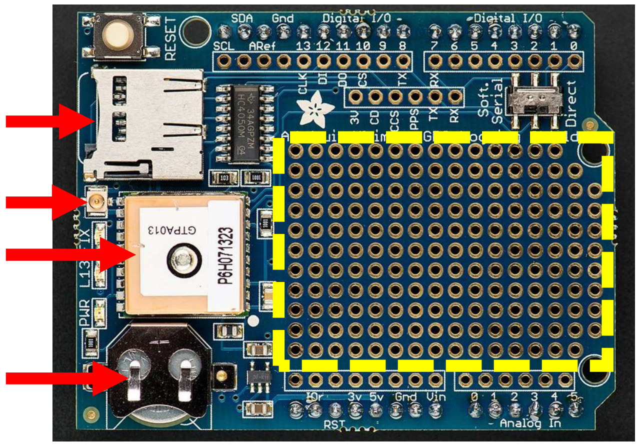


Figure 2: Shown is the Adafruit Ultimate GPS Logger Shield. From top to bottom, the red arrows show the microSD socket, the external antenna connector, the GPS unit, and the coin cell battery holder. The yellow dashed square designates the prototyping area

Antennas

- The GPS shield has a built-in patch antenna. For the MegaSat, this antenna will suffice
- Payload shielding or sensitivity requirements may require an external antenna. See Figure 3 for where this would be attached
  - Most GPS antennas will have an SMA connector, so a SMA to uFL connector will be required
- The GPS shield detects external antennas and uses them automatically
- NMEA sentences provide antenna status (internal, external, or that a problem exists)

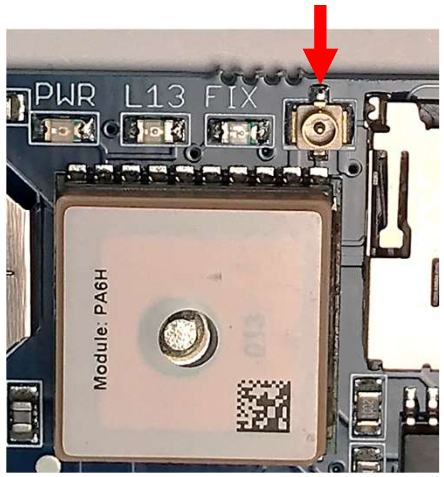


Figure 3: The red arrow is pointing to the Adafruit Ultimate GPS Logger Shield's external antenna connector. It is located between the microSD socket and the FIX LED. To connect an antenna, a uFL to SMA connector will be required

LaACES Student Ballooning Course

Real Time Clock (RTC)

- A Real Time Clock (RTC) is a computer clock that keeps the current time
- When the GPS has a fix, the RTC sets itself to the current UTC time
  - If the GPS loses power, this ensures the GPS will keep the correct time
- The GPS has a built-in RTC, but not a built-in battery. A coin cell battery needs to be inserted. See Figure 4
- A new battery can run the RTC for 7+ years

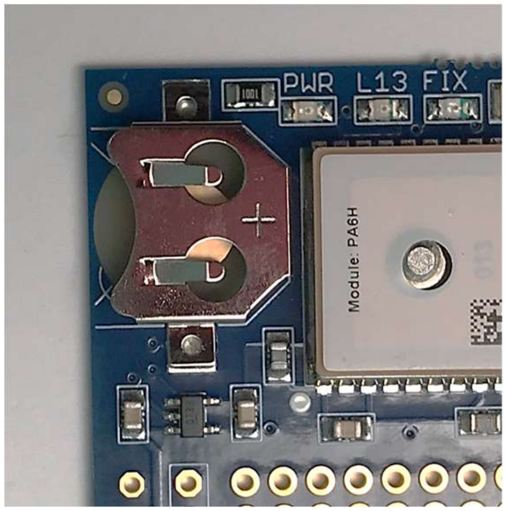


Figure 4: Shown is the Adafruit Ultimate GPS Logger Shield’s coin cell battery holder. When inserting a coin cell battery, ensure the positive side is facing up, away from the Shield

LaACES Student Ballooning Course

LEDs on GPS Shield

- There are 3 different LEDs on the GPS shield, see Figure 5
  - Green (PWR)
    - Power LED. If it's not on, the shield doesn't have power
  - Yellow (L13, SD card access)
    - Solid when Arduino is bootloading
    - Flickers when SD card is being accessed
  - Red (FIX)
    - Blinking every second, GPS does not have a fix
    - Blinking once every 15 seconds, GPS has a fix

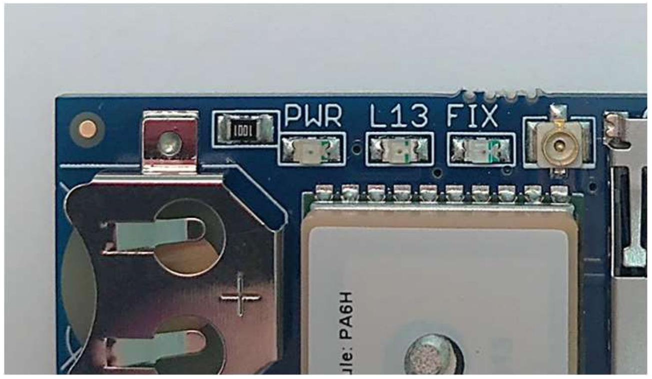


Figure 5: Shown are the three LEDs on the Adafruit Ultimate GPS Logger Shield. They are located next to the external antenna connector. From left to right, there is the power LED (lit when the Shield has power), the L13 LED (lit when the Arduino is bootloading and flickers when the SD card used), and the FIX LED (blinks at different frequencies corresponding to the GPS fix quality)

LaACES Student Ballooning Course

Prototyping and Breakout Areas

- There is a prototyping area on the GPS shield, see Figure 6
- Next to the Prototyping area are six breakouts
    - 3V: Outputs 3 volts
    - CD: Card-detect output from the microSD socket. It is shorted to ground if there is not a card inserted
    - CCS: Card chip select line. It is connected to digital 10 by default
    - PPS: This is a pulse per second output when the GPS has a fix
    - TX: GPS transmit line
    - RX: GPS receive line

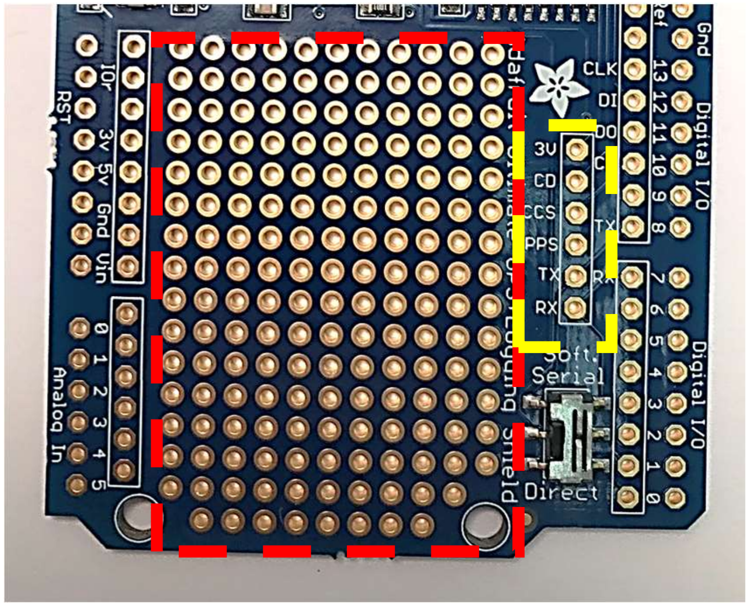


Figure 6: Shown is the prototyping and breakout areas on the Adafruit Ultimate GPS Logger Shield. The red dashed box is surrounding the prototyping area. The yellow dashed box surrounds the breakout area. The RX and TX breakout pins are used for Software Serial communication

Communication: Two Methods

- Two ways to communicate with the GPS shield: Direct Connect and Software Serial
- Direct Connect connects the Shield to the Arduino’s Serial 0 channel. This is the channel used for uploading sketches
    - The Arduino is removed from communication. Commands sent directly to the Shield using the Serial Monitor
- Software Serial uses jumpers to utilize a different Serial channel so sketches can be uploaded to the Arduino

LaACES Student Ballooning Course

Communication: Direct Connect

- Direct Connect connects the GPS serial TTL UART directly to the Arduino's USB-serial converter chip
- To enable Direct Connect:
  - Start with the switch on Soft Serial to upload a blank sketch
  - Then flip the switch down (Figure 7)
  - NMEA sentences will start showing up on the Serial Monitor
- **While using Direct Connect, you cannot upload sketches to the Arduino!** The programming UART is being used by the GPS shield

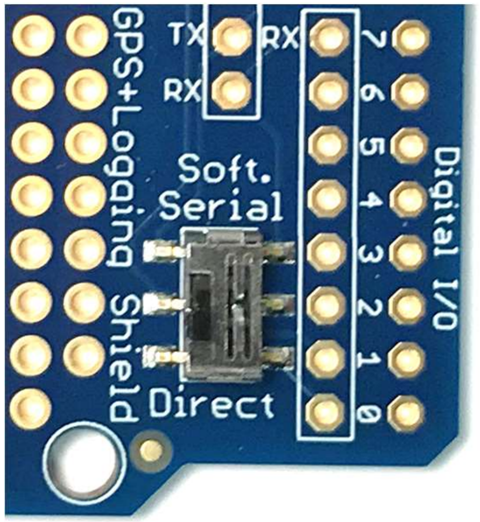


Figure 7: Shown is the communication switch on the Adafruit Ultimate GPS Logger Shield. In this position, the Shield is communicating via Direct Connect. This method has the Shield using the main Arduino UART to communicate with the computer. Because of this, sketches cannot be uploaded while using Direct Connect

LaACES Student Ballooning Course

Communication: Software Serial

- Software Serial allows sketches to be uploaded using the Arduino's main UART because a different UART is used for GPS communication
- The Shield switch should be on Soft Serial. Jumpers will be used to connect GPS TX and RX pins to two Arduino UART pins. See Figure 8
- To communicate with the GPS shield, upload sketches to the Arduino. Adafruit has a GPS library to help receive and parse the GPS data stream

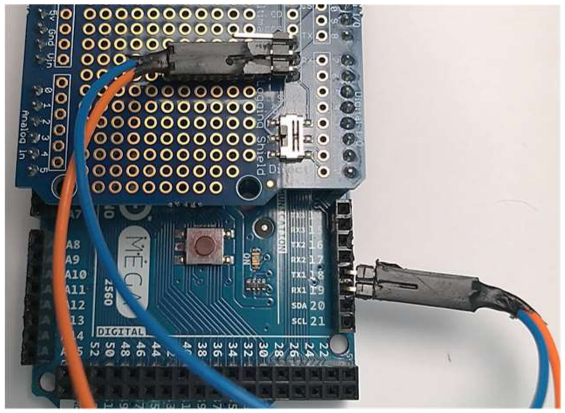


Figure 8: Shown is the communication switch on the Adafruit Ultimate GPS Logger Shield and jumpers connecting the RX and TX breakout pins to Arduino pins 18 and 19, respectively. When the switch is in this position, the Shield is communicating via Software Serial. This method allows sketches to be uploaded to the Arduino

LaACES Student Ballooning Course

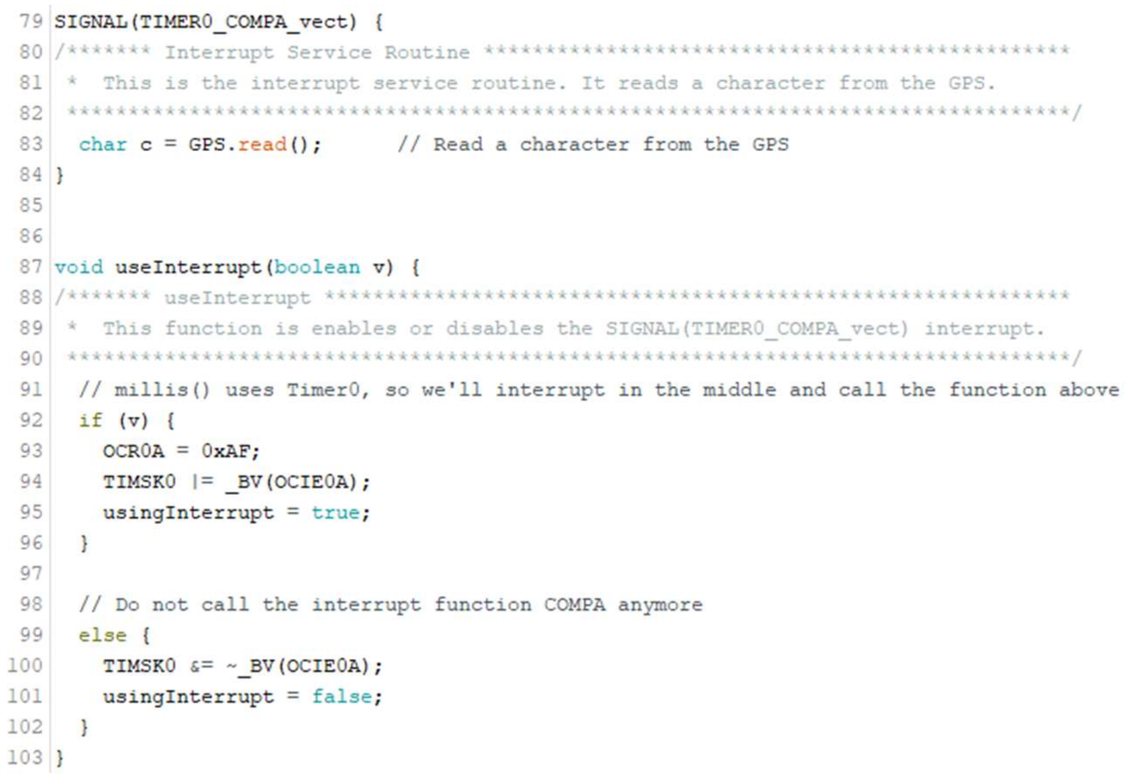
Software Serial: Interrupt

- GPS send data to a Serial buffer. This buffer needs to be read manually. There are two ways – polling and interrupts. Interrupts are the proper way.
- Polling
  - Arduino checks to see if anything is in the serial buffer on a regular interval. Must stop what it is doing to check.
    - Constantly checking the front door to see if the Amazon package is delivered yet
- Interrupt
  - When a specific signal is received, the Arduino goes and reads the serial buffer. Otherwise, it keeps on doing other things.
    - Cleaning the house, doing laundry, etc. until the doorbell rings, then the front door is checked for the Amazon package.

LaACES Student Ballooning Course

Interrupt Example

- Figure 9 is the code used to interrupt and read the GPS serial buffer
- Timer0 is an internal timer used for millis(). Interrupt between the millis() pulses to read a character from the serial buffer every millisecond.



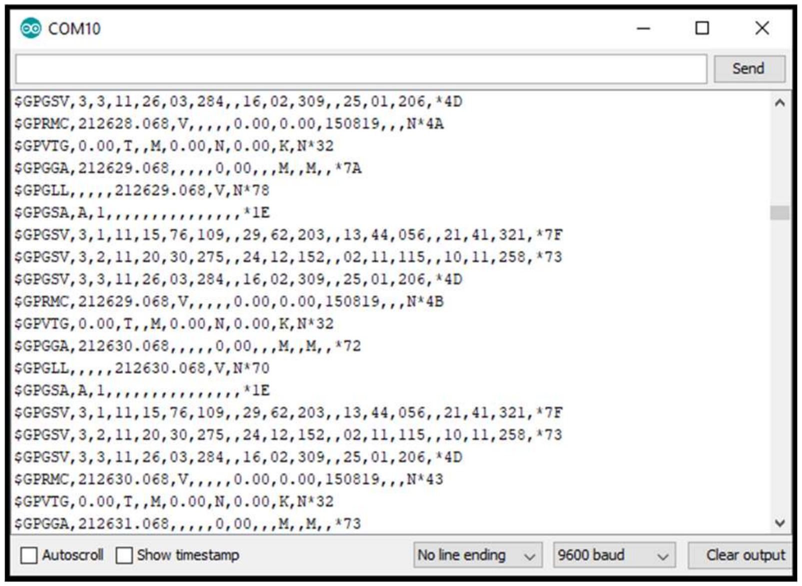
```
SIGNAL(TIMER0_COMPA_vect) {
/******* Interrupt Service Routine **************************************************
 * This is the interrupt service routine. It reads a character from the GPS.
 ***************************************************************************************/
  char c = GPS.read();      // Read a character from the GPS
}


void useInterrupt(boolean v) {
/******* useInterrupt ******************************************************************
 * This function is enables or disables the SIGNAL(TIMER0_COMPA_vect) interrupt.
 ***************************************************************************************/
  // millis() uses Timer0, so we'll interrupt in the middle and call the function above
  if (v) {
    OCR0A = 0xAF;
    TIMSK0 |= _BV(OCIE0A);
    usingInterrupt = true;
  }

  // Do not call the interrupt function COMPA anymore
  else {
    TIMSK0 &= ~_BV(OCIE0A);
    usingInterrupt = false;
  }
}
```

Figure 9: The top function simply reads a character from the GPS serial buffer. The second function uses Timer0 (an internal timer on the Mega), which is used to increment the millis() call. This function says to interrupt between the millis() increments (so as not to interfere with it). This will result in the Mega reading a character from the GPS serial buffer once a millisecond.

LaACES Student Ballooning Course

Data Streams

- Data received from the GPS shield will be NMEA sentences.
  - Figure 10 shows what raw GPS data will look like in the Serial Monitor
- $GPRMC,212628.068,V,,,,,0.00,0.00,150819,,,N*4A
  - This is an example of a NMEA sentence pulled from Figure 10
  - From this data stream, we can see it was taken on August 15, 2019 at 21:26:28 UTC. A fix was not achieved.



```
$GPGSV,3,3,11,26,03,284,,16,02,309,,25,01,206,*4D
$GPRMC,212628.068,V,,,,,0.00,0.00,150819,,,N*4A
$GPVTG,0.00,T,,M,0.00,N,0.00,K,N*32
$GPGGA,212629.068,,,,,0,00,,,M,,M,,*7A
$GPGLL,,,,,212629.068,V,N*78
$GPGSA,A,1,,,,,,,,,,,,,,,*1E
$GPGSV,3,1,11,15,76,109,,29,62,203,,13,44,056,,21,41,321,*7F
$GPGSV,3,2,11,20,30,275,,24,12,152,,02,11,115,,10,11,258,*73
$GPGSV,3,3,11,26,03,284,,16,02,309,,25,01,206,*4D
$GPRMC,212629.068,V,,,,,0.00,0.00,150819,,,N*4B
$GPVTG,0.00,T,,M,0.00,N,0.00,K,N*32
$GPGGA,212630.068,,,,,0,00,,,M,,M,,*72
$GPGLL,,,,,212630.068,V,N*70
$GPGSA,A,1,,,,,,,,,,,,,,,*1E
$GPGSV,3,1,11,15,76,109,,29,62,203,,13,44,056,,21,41,321,*7F
$GPGSV,3,2,11,20,30,275,,24,12,152,,02,11,115,,10,11,258,*73
$GPGSV,3,3,11,26,03,284,,16,02,309,,25,01,206,*4D
$GPRMC,212630.068,V,,,,,0.00,0.00,150819,,,N*43
$GPVTG,0.00,T,,M,0.00,N,0.00,K,N*32
$GPGGA,212631.068,,,,,0,00,,,M,,M,,*73
```

☐ Autoscroll ☐ Show timestamp   No line ending   9600 baud   Clear output

Figure 10: Shown is an example of the raw output of the Adafruit Ultimate GPS Logger Shield on the serial monitor when the Shield is switched to Direct Connect. The GPS did not have a fix, so most data fields are blank

Parsing

- When using Software Serial, the NMEA data can be parsed using the Adafruit GPS library. This allows information (such as time, date, latitude, speed, etc.) to be easily available
- Adafruit_GPS.h file in library lists all possible information obtained from parsing
- $GPRMC,212628.068,V,,,,,0.00,0.00,150819,,,N*4A

GPS Command	Result
GPS.day	15
GPS.month	8
GPS.hour	21
GPS.second	28

LaACES Student Ballooning Course

Information and Image References

- Information
  - https://learn.adafruit.com/adafruit-ultimate-gps-logger-shield/overview
- Images
  - GPS shield – slides 2 and 3: https://www.adafruit.com/product/1272?gclid=Cj0KCQjwhdTqBRDNARIsABsOl9_UNhRtd2VDGjp-aaau_JLLP2AxSAH72n2Fm4Md7Ipr6MLFVqatdUYaAlb1EALw_wcB