Yet another tracking device - Hardware

Created by Unknown User (danmi), last modified on Jan 21, 2016

Materials

UNO R3 MEGA328P Arduino Compatible development board $6.99 usd

Sim900 GPRS/GSM Arduino Shield $19.62 usd. If you look to save even more money, and don't mind some simple wire soldering please use this chip SIM800L for $6.97 usd.

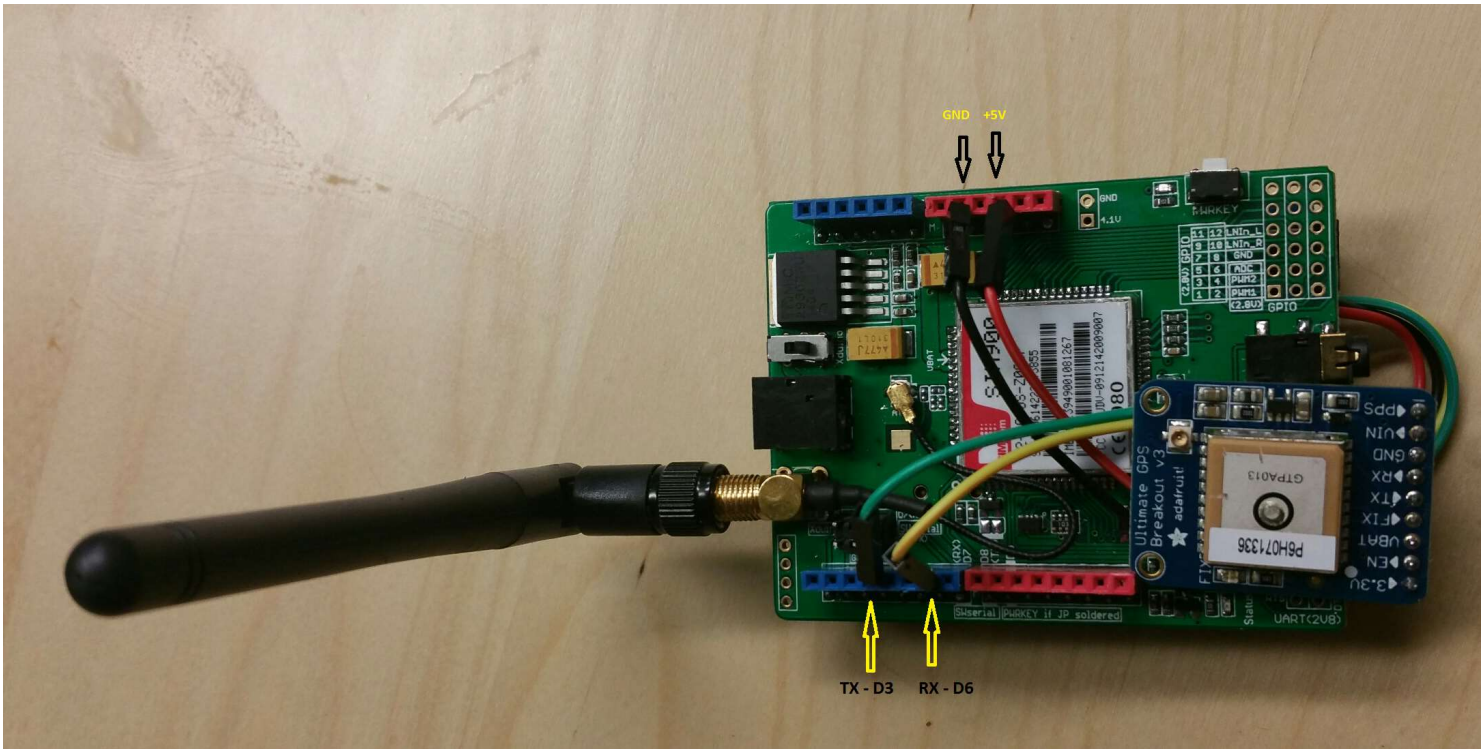
UBlox NEO-6M GPS Module $10.40 usd

Assembly

Add a phone SIM card (I used a Rogers $5cad/month data only 10Mb) to the SIM900 hardware. Assembly the SIM900 shield together with the Arduino controller.

Set the serial port jumpers to use D7-D8. See more details here http://www.seeedstudio.com/wiki/GPRS_Shield_V1.0#Pins_usage_on_Arduino

Run 4 wires for the GPS. +5 and GND and also Rx to D6, Tx to D3.



Code

On your PC install the Arduino studio from here then upload the following code into the Arduino board; In the code don't forget to change the APN string for your own provider and the URL to where you host the software.

Disconnect the board from the PC, provide a battery and start tracking!

YetAnotherTrackingDevice

```
/*
Dan Mincu - JSI TELECOM @ 2016
this software uses Adafruit_GPS.
Documentation https://learn.adafruit.com/downloads/pdf/adafruit-ultimate-gps.pdf
Library here https://github.com/adafruit/Adafruit-GPS-Library
If you use another GPS module you need to parse the lat and long differntly
*/
#include <Adafruit_GPS.h>
#include <SoftwareSerial.h>
SoftwareSerial myGPSSerial(3, 6);
SoftwareSerial mySerial(7, 8);
Adafruit_GPS GPS(&myGPSSerial);

// Set GPSECHO to 'false' to turn off echoing the GPS data to the Serial console
// Set to 'true' if you want to debug and listen to the raw GPS sentences.
#define GPSECHO  false
```

```
// this keeps track of whether we're using the interrupt
// off by default!
boolean usingInterrupt = false;
void useInterrupt(boolean); // Func prototype keeps Arduino 0023 happy
void setup()
{
  pinMode(13, OUTPUT);
  Serial.begin(115200);
  Serial.println("Yet another tracking device!");
  // 9600 NMEA is the default baud rate for Adafruit MTK GPS's- some use 4800
  GPS.begin(9600);
  mySerial.begin(19200);// the GPRS baud rate

  // uncomment this line to turn on RMC (recommended minimum) and GGA (fix data) including altitude
  GPS.sendCommand(PMTK_SET_NMEA_OUTPUT_RMCGGA);
  // uncomment this line to turn on only the "minimum recommended" data
  //GPS.sendCommand(PMTK_SET_NMEA_OUTPUT_RMCONLY);
  // For parsing data, we don't suggest using anything but either RMC only or RMC+GGA since
  // the parser doesn't care about other sentences at this time

  // Set the update rate
  GPS.sendCommand(PMTK_SET_NMEA_UPDATE_1HZ);   // 1 Hz update rate
  // For the parsing code to work nicely and have time to sort thru the data, and
  // print it out we don't suggest using anything higher than 1 Hz
  // Request updates on antenna status, comment out to keep quiet
  GPS.sendCommand(PGCMD_ANTENNA);
  // the nice thing about this code is you can have a timer0 interrupt go off
  // every 1 millisecond, and read data from the GPS for you. that makes the
  // loop code a heck of a lot easier!
  useInterrupt(true);
  delay(1000);
  // Ask for firmware version
  mySerial.println(PMTK_Q_RELEASE);
}

// Interrupt is called once a millisecond, looks for any new GPS data, and stores it
SIGNAL(TIMER0_COMPA_vect) {
  char c = GPS.read();
  // if you want to debug, this is a good time to do it!
#ifdef UDR0
  if (GPSECHO)
    if (c) UDR0 = c;
    // writing direct to UDR0 is much much faster than Serial.print
    // but only one character can be written at a time.
#endif
}
void useInterrupt(boolean v) {
  if (v) {
    // Timer0 is already used for millis() - we'll just interrupt somewhere
    // in the middle and call the "Compare A" function above
    OCR0A = 0xAF;
    TIMSK0 |= _BV(OCIE0A);
    usingInterrupt = true;
  } else {
    // do not call the interrupt function COMPA anymore
    TIMSK0 &= ~_BV(OCIE0A);
    usingInterrupt = false;
  }
}
uint32_t timer = millis();
void loop()                     // run over and over again
{
  myGPSSerial.listen();
  // in case you are not using the interrupt above, you'll
  // need to 'hand query' the GPS, not suggested :(
  if (! usingInterrupt) {
    // read data from the GPS in the 'main loop'
    char c = GPS.read();
    // if you want to debug, this is a good time to do it!
    if (GPSECHO)
      if (c) Serial.print(c);
  }

```

```
  // if a sentence is received, we can check the checksum, parse it...
  if (GPS.newNMEAreceived()) {
    // a tricky thing here is if we print the NMEA sentence, or data
    // we end up not listening and catching other sentences!
    // so be very wary if using OUTPUT_ALLDATA and trytng to print out data
    //Serial.println(GPS.lastNMEA());   // this also sets the newNMEAreceived() flag to false

    if (!GPS.parse(GPS.lastNMEA()))   // this also sets the newNMEAreceived() flag to false
      return;  // we can fail to parse a sentence in which case we should just wait for another
  }
  // if millis() or timer wraps around, we'll just reset it
  if (timer > millis())  timer = millis();
  // approximately every 20 seconds or so
  if (millis() - timer > 20000) {

    Serial.print("\nTime: ");
    Serial.print(GPS.hour, DEC); Serial.print(':');
    Serial.print(GPS.minute, DEC); Serial.print(':');
    Serial.print(GPS.seconds, DEC); Serial.print('.');
    Serial.println(GPS.milliseconds);
    Serial.print("Date: ");
    Serial.print(GPS.day, DEC); Serial.print('/');
    Serial.print(GPS.month, DEC); Serial.print("/20");
    Serial.println(GPS.year, DEC);
    Serial.print("Fix: "); Serial.print((int)GPS.fix);
    Serial.print(" quality: "); Serial.println((int)GPS.fixquality);

    if (GPS.fix) {

      Serial.print("Location (in degrees): ");
      Serial.print(GPS.latitudeDegrees, 6);
      Serial.print(", ");
      Serial.println(GPS.longitudeDegrees, 6);
      String latitude = String(GPS.latitudeDegrees, 7);
      String longitude = String(GPS.longitudeDegrees, 7);
      Serial.print("Speed (knots): "); Serial.println(GPS.speed);
      Serial.print("Angle: "); Serial.println(GPS.angle);
      Serial.print("Altitude: "); Serial.println(GPS.altitude);
      Serial.print("Satellites: "); Serial.println((int)GPS.satellites);

      // don't forget to call your own Web API end point here
      String url = "AT+HTTPPARA=\"URL\",\"http://danix.cloudapp.net/yatd/AddTrackpoint?device=D1&lat=" + latitude + "&lon="+ longitude
      SubmitHttpRequest(url);
      timer = millis(); // reset the timer
    }
  }
}

///SubmitHttpRequest()
///this function is submit a http request
///attention:the time of delay is very important, it must be set enough
void SubmitHttpRequest(String url)
{
  mySerial.listen();
  delay(100);

  mySerial.println("AT+CSQ");
  delay(100);

  ShowSerialData();// this code is to show the data from gprs shield, in order to easily see the process of how the gprs shield submit

  mySerial.println("AT+CGATT?");
  delay(100);

  ShowSerialData();

  mySerial.println("AT+SAPBR=3,1,\"CONTYPE\",\"GPRS\"");//setting the SAPBR, the connection type is using gprs
  delay(1000);

  ShowSerialData();
  // this APN is good for Rogers Canada, change it accordingly
  mySerial.println("AT+SAPBR=3,1,\"APN\",\"internet.com\"");//setting the APN, the second need you fill in your local apn server
  delay(4000);
```

```
164
165     ShowSerialData();
166
167     mySerial.println("AT+SAPBR=1,1");//setting the SAPBR, for detail you can refer to the AT command mamual
168     delay(2000);
169
170     ShowSerialData();
171
172     mySerial.println("AT+HTTPINIT"); //init the HTTP request
173
174     delay(2000);
175     ShowSerialData();
176
177     mySerial.println(url);
178     delay(1000);
179
180     ShowSerialData();
181
182     mySerial.println("AT+HTTPACTION=0");//submit the request
183     delay(10000);//the delay is very important, the delay time is base on the return from the website, if the return datas are very large
184
185     ShowSerialData();
186
187     mySerial.println("AT+HTTPREAD");// read the data from the website you access
188     delay(300);
189
190     ShowSerialData();
191
192     mySerial.println("");
193     delay(100);
194   }
195
196   void ShowSerialData()
197   {
198     while(mySerial.available()!=0)
199     {
200       for (int i=0; i < 5; i++)
201       {
202         // flicker the board control LED when there is communication with the GPRS shield
203         digitalWrite(13, HIGH);
204         delay(2);
205         digitalWrite(13, LOW);
206         delay(2);
207       }
208       Serial.write(mySerial.read());
209     }
   }
```

No labels