

 adafruit learning system

## GPS Dog Collar

Created by Becky Stern



# Guide Contents

Guide Contents	2
Overview & Parts	3
Code & Wiring	5
Downloads	9

## Overview & Parts

---

Wonder if you're giving your dog the exercise she needs? Like electronics? Make your own GPS dog collar!

You will need:

- [Adafruit Ultimate GPS Breakout \(http://adafru.it/746\)](http://adafru.it/746) - 66 channel w/10 Hz updates - MTK3339 chipset
- [Atmega32u4 Breakout Board \(http://adafru.it/296\)](http://adafru.it/296)
- [Monochrome 128x32 OLED graphic display \(http://adafru.it/661\)](http://adafru.it/661)
- [3 x AAA Battery Holder \(http://adafru.it/727\)](http://adafru.it/727) and batteries
- stranded hookup wire
- wire-edged ribbon
- scrap of fabric
- electrical tape or dark nail polish (optional)

Tools:

- Soldering iron
- wire strippers
- pliers
- scissors
- needle and thread
- iron
- sewing pins
- sewing machine (optional)



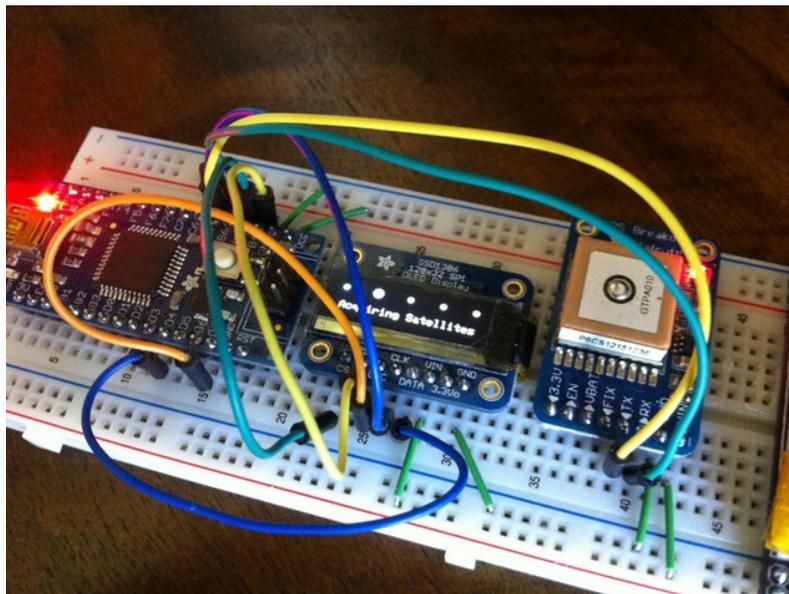
The progress bar fills up as you stroll around the neighborhood. Set your goal ahead of time and watch the miles rack up.

## Code & Wiring

When Becky and I started this dog collar project, Arduino 1.0.1 had just come out with support for the Arduino Leonardo. I had an Adafruit 32u4 breakout board on hand, and thought it would be a good way to test it out with the official Leonardo bootloader. I used a USBtinyISP and the Arduino software to burn the Leonardo to the 32u4 breakout board without any issues.

My first step was getting the tiny, and awesome Adafruit Monochrome 128x32 OLED to work with the Leonardo software.

[Download the latest code on Github \(http://adafru.it/aje\)](http://adafru.it/aje)



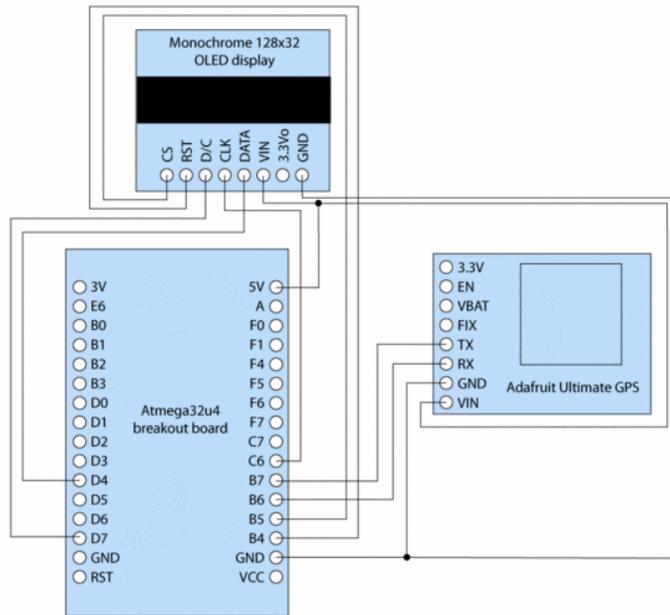
The only real issue I ran into was matching up the pins from the Atmega32u4 to the pins in the code examples/libraries for the OLED display. I ended up creating a pin reference sheet similar to [this one by Johngineer \(which is much better than mine\)](http://adafru.it/ajf). (<http://adafru.it/ajf>)

At the top of the code, I have comments showing what OLED pins match up to which pins on the 32u4 breakout.

```
#define OLED_DC 6 //Atmega32u4 Breakout Pin D7
#define OLED_CS 9 //Atmega32u4 Breakout Pin B5
#define OLED_CLK 5 //Atmega32u4 Breakout Pin C6
#define OLED_MOSI 4 //Atmega32u4 Breakout Pin D4 (Pin on OLED labeled DATA)
#define OLED_RESET 8 //Atmega32u4 Breakout Pin B4
```

I also show where to connect up the Adafruit Ultimate GPS module.

```
//Connect GPS TX to Atmega32u4 Breakout Pin B7 (Leonardo Pin D10)
//Connect GPS RX to Atmega32u4 Breakout Pin B6 (Leonardo Pin D11)
```



If you scroll down a bit in the code, you will see where you enter the total distance you would like your dog to walk in a day. I have set the default to 3 miles.

```
//ENTER YOUR DESIRED DISTANCE GOAL (IN MILES)
//-----
float GOAL = 3; //Distances can include decimal points
//-----
```

I used the [TinyGPS library \(http://adafru.it/ajg\)](http://adafru.it/ajg) to do most of the heavy lifting, and pulled a lot of code from my [Coobro Geo \(http://adafru.it/652\)](http://adafru.it/652) code. The heart of the code is all about taking constant distance measurements. Every time the code loops, it looks at where you were, and where you are.

```
unsigned long calc_dist(float flat1, float flon1, float flat2, float flon2)
{
  float dist_calc=0;
  float dist_calc2=0;
  float diflat=0;
  float diflon=0;

  diflat=radians(flat2-flat1);
  flat1=radians(flat1);
  flat2=radians(flat2);
  diflon=radians((flon2)-(flon1));

  dist_calc = (sin(diflat/2.0)*sin(diflat/2.0));
  dist_calc2= cos(flat1);
  dist_calc2*=cos(flat2);
  dist_calc2*=sin(diflon/2.0);
  dist_calc2*=sin(diflon/2.0);
  dist_calc +=dist_calc2;
```

```

dist_calc=(2*atan2(sqrt(dist_calc),sqrt(1.0-dist_calc)));

dist_calc*=6371000.0; //Converting to meters
return dist_calc;
}

```

When standing still, the GPS coordinates will jump around slightly. I didn't want this to affect the total distance traveled, so I had the code first make sure you were moving. If you are moving, it adds the distance value from the code above to a running total to determine your total distance traveled.

```

if (gps.f_speed_kmph() > 3.9)
{
  if (start == 1)
  {
    start = 0;
    lastFlat = flat;
    lastFlon = flon;
  }
  else
  {
    //totalDistance = gps.distance_between(flat, flon, LONDON_LAT, LONDON_LON);
    totalDistance = totalDistance + calc_dist(flat, flon, lastFlat, lastFlon);
    lastFlat = flat;
    lastFlon = flon;
  }
}
}

```

My favorite part of coding this project was making that tiny OLED display useful information that would be easy to see and understand at a quick glance. I started by creating a neat 'Acquiring Satellites' animation when you first turn on the device. I then created a nice bar graph that shows your progress towards your goal. Above the bar graph is a running mileage counter. All of these numbers can easily be converted to km if needed.

```

display.clearDisplay();

display.setTextSize(1);
display.setTextColor(WHITE);
display.setCursor(0,0);

float fDist = totalDistance;
//convert meters to miles
fDist *= 0.000621371192;
//float fSpeed = gps.f_speed_kmph();
printLCDFloat(fDist, 2);
display.print(" Miles (");

float targetDist = fDist / GOAL;

printLCDFloat(targetDist*100, 0);
display.print("%)");

```

```

display.drawLine(0, 12, 0, 31, WHITE);

display.drawLine(63, 28, 63, 31, WHITE);
display.drawLine(127, 12, 127, 31, WHITE);
display.drawLine(31, 28, 31, 31, WHITE);

display.drawLine(95, 28, 95, 31, WHITE);
display.drawLine(0, 28, 127, 28, WHITE);
display.drawLine(0, 12, 127, 12, WHITE);

display.fillRect(2, 14, (124 * targetDist), 13, 1);

if (gps.hdop() > 2000) {
  //display.fillRect(2, 14, (124), 13, BLACK);
  display.fillRect(0, 0, 128, 32, BLACK);
  display.fillCircle(6, 6, 2, WHITE);
  display.fillCircle(64, 6, 2, WHITE);
  display.fillCircle(122, 6, 2, WHITE);
  display.fillCircle(35, 6, 2, WHITE);
  display.fillCircle(93, 6, 2, WHITE);

  if (i==0){
    display.drawCircle(6, 6, 4, WHITE);
  }
  if (i==1){
    display.drawCircle(35, 6, 4, WHITE);
  }
  if (i==2){
    display.drawCircle(64, 6, 4, WHITE);
  }
  if (i==3){
    display.drawCircle(93, 6, 4, WHITE);
  }
  if (i==4){
    display.drawCircle(122, 6, 4, WHITE);
    i = 0;
  } else {
    i++;
  }

  display.setTextColor(WHITE);
  display.setCursor(5,20);
  display.print("Acquiring Satellites");
}

display.display();

```

That about sums up the main parts of the code. Your challenge, should you choose to accept, is to improve upon what we have done here. The next logical step would be to use the built in data logger feature of the Ultimate GPS module, and map your dog walks when you get home.

## Downloads

---

Download the latest code on Github (<http://adafru.it/aje>)