LS20031 GPS Assembly Guide

by followr | June 03, 2010 | 48 comments Skill Level: ★ Beginner

Welcome

Congratulations on your purchase of a Locosys LS20031 GPS module which can tell you where you are five times a second! Where are you now? How about now? I bet you don't know and that's why you need to get your LS20031 working. So let's get started...

The LS20031 module does not have a connector attached so in order get data from the unit we need to attach a connector or some connecting wires. Whatever method of connection you use, remember it is important to keep the square ceramic antenna so it is pointed toward the sky.

Preparation

Tools

You will need the following tools and supplies:

- Soldering iron (e.g. TOL-00085)
- Solder (e.g. TOL-09161)
- Third Hand (e.g. TOL-09317) (optional)

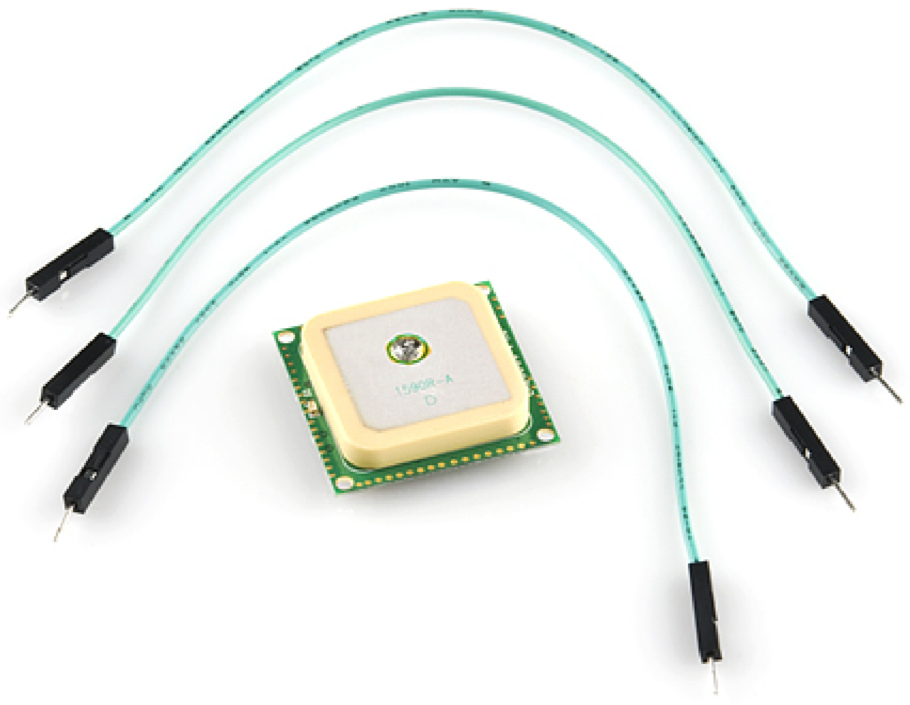
Parts

For this assembly guide we will be using the following parts as shown below:

- Locosys LS20031 GPS module
- Break Away Headers - Right Angle (5 pins wide)

You may prefer to use plain wire or jumpers like those pictured below but we will not cover this method here:

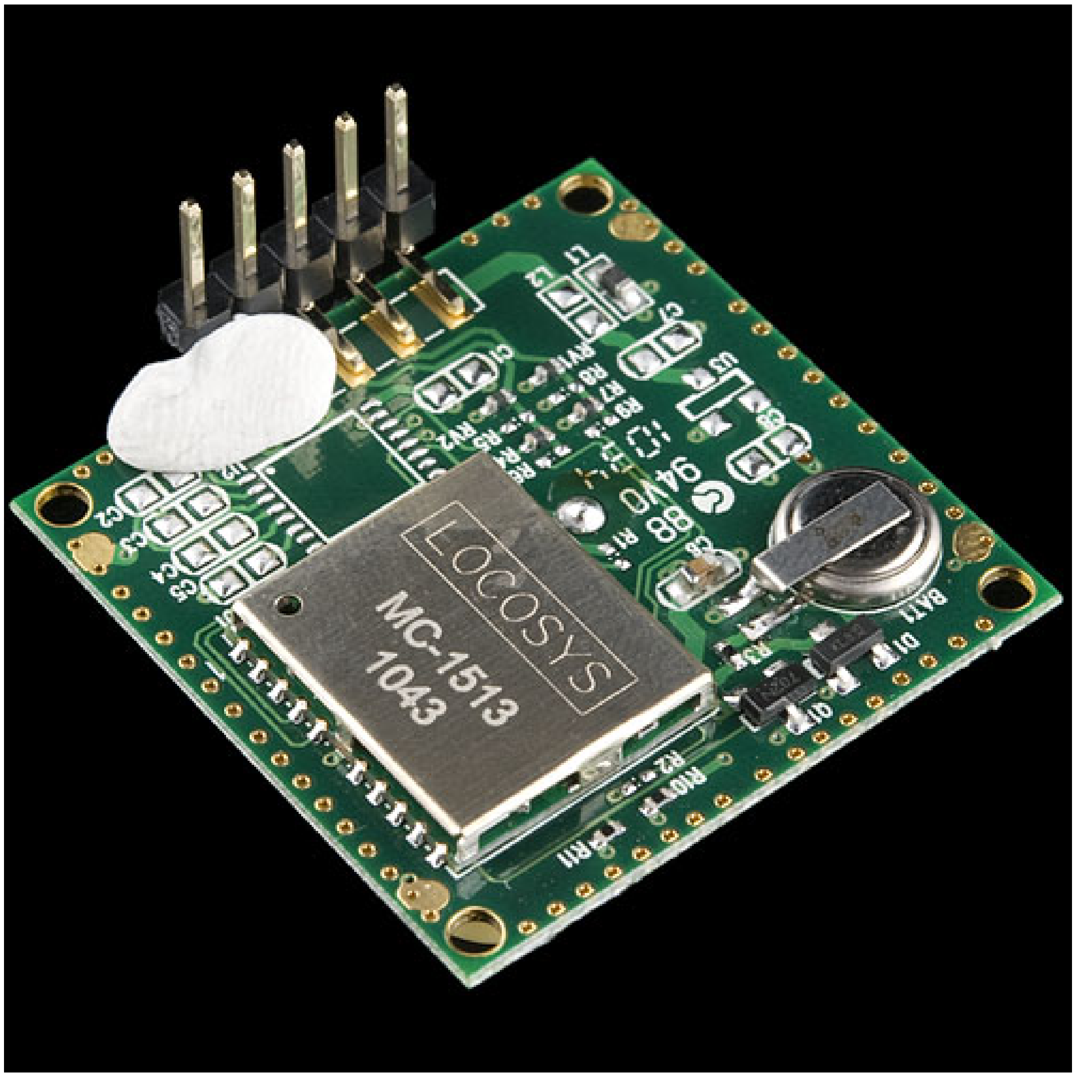
- Jumper wires



Assembly

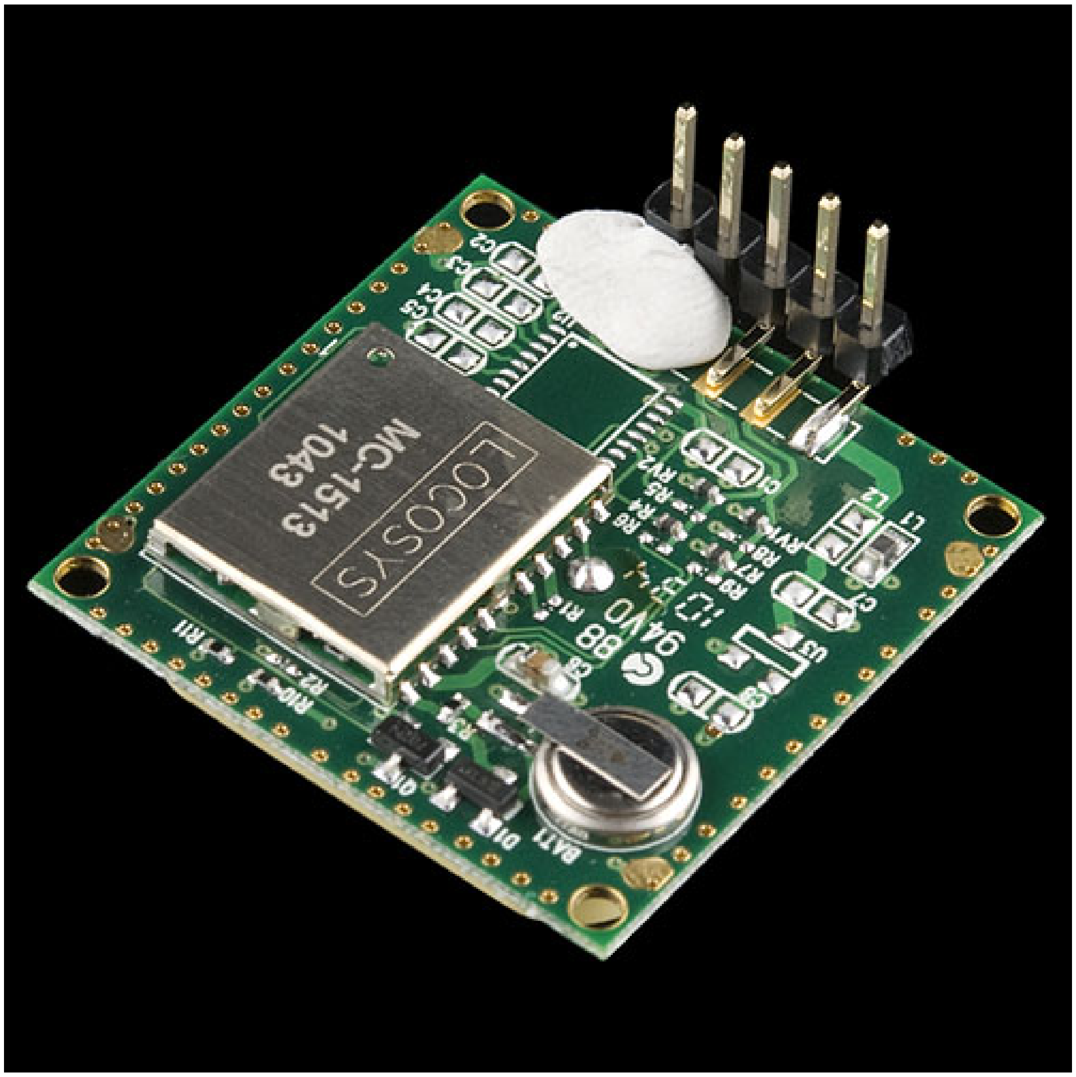
The most difficult part of the assembly process is probably keeping the right-angle header connector in place before you have soldered any of its pins. The third arm with the two clips works very well for this. If you're really good you might be able to hold the connector in place with pliers while you solder.

Temporarily fasten one of the outside pins of the connector in place ensuring that all the connector pins overlap the appropriate gold-colored pads on the GPS module circuit board. You will notice the small holes (vias) around the outside of the module run under two of the pins—it is okay if these two pins of the connector make contact with the small holes as in this case both the pins and vias are connected to ground.



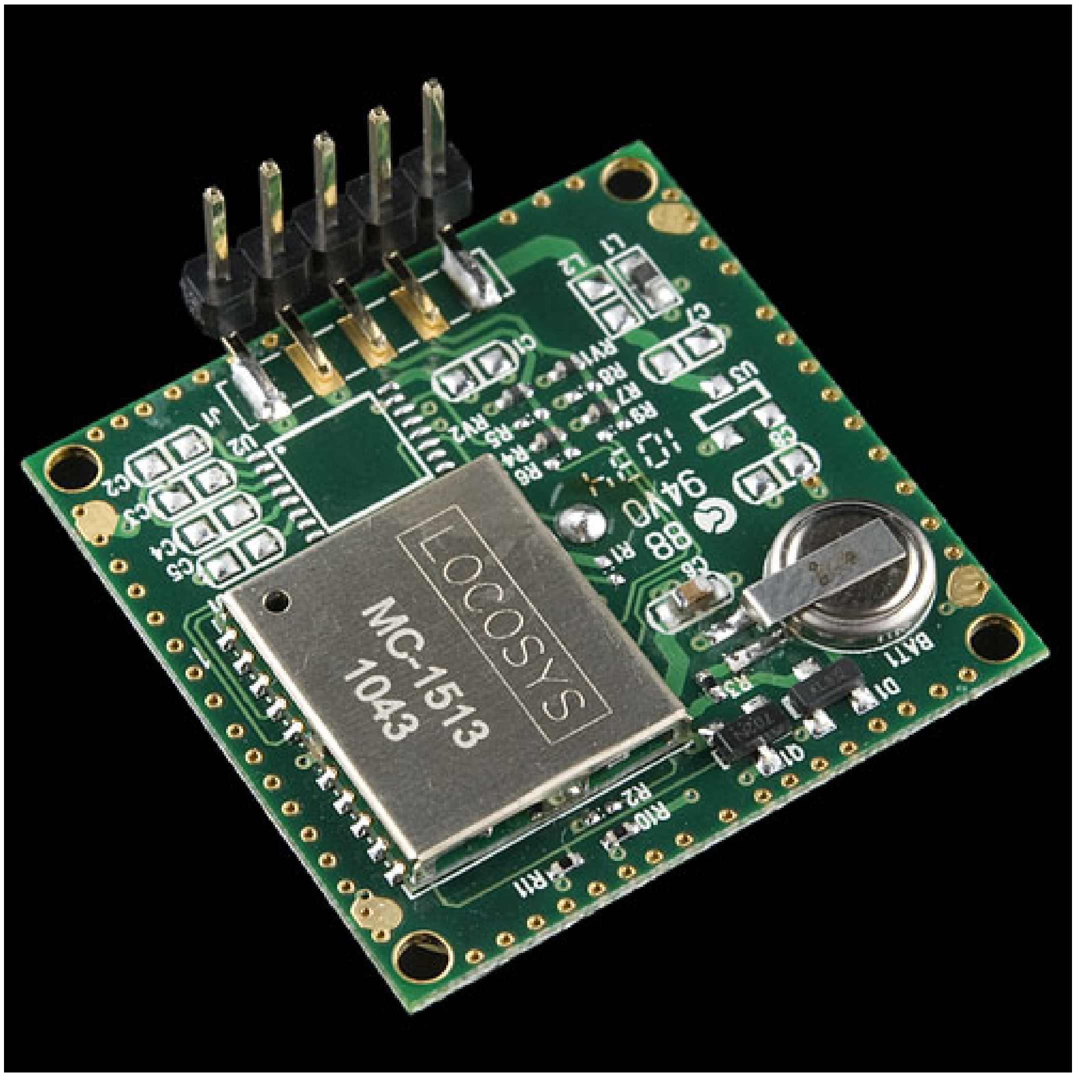
Once you're satisfied with the position of the connector, solder the pin at the opposite end of the connector. You can use more solder than you might normally as it will need to provide structural support in addition to electrical contact. As usual try not to hold the soldering iron in contact for more than a few seconds. Also make sure you don't accidentally de-solder any of the other components on the board or flick solder onto the board.

When you've soldered the first pin it should look similar to this:



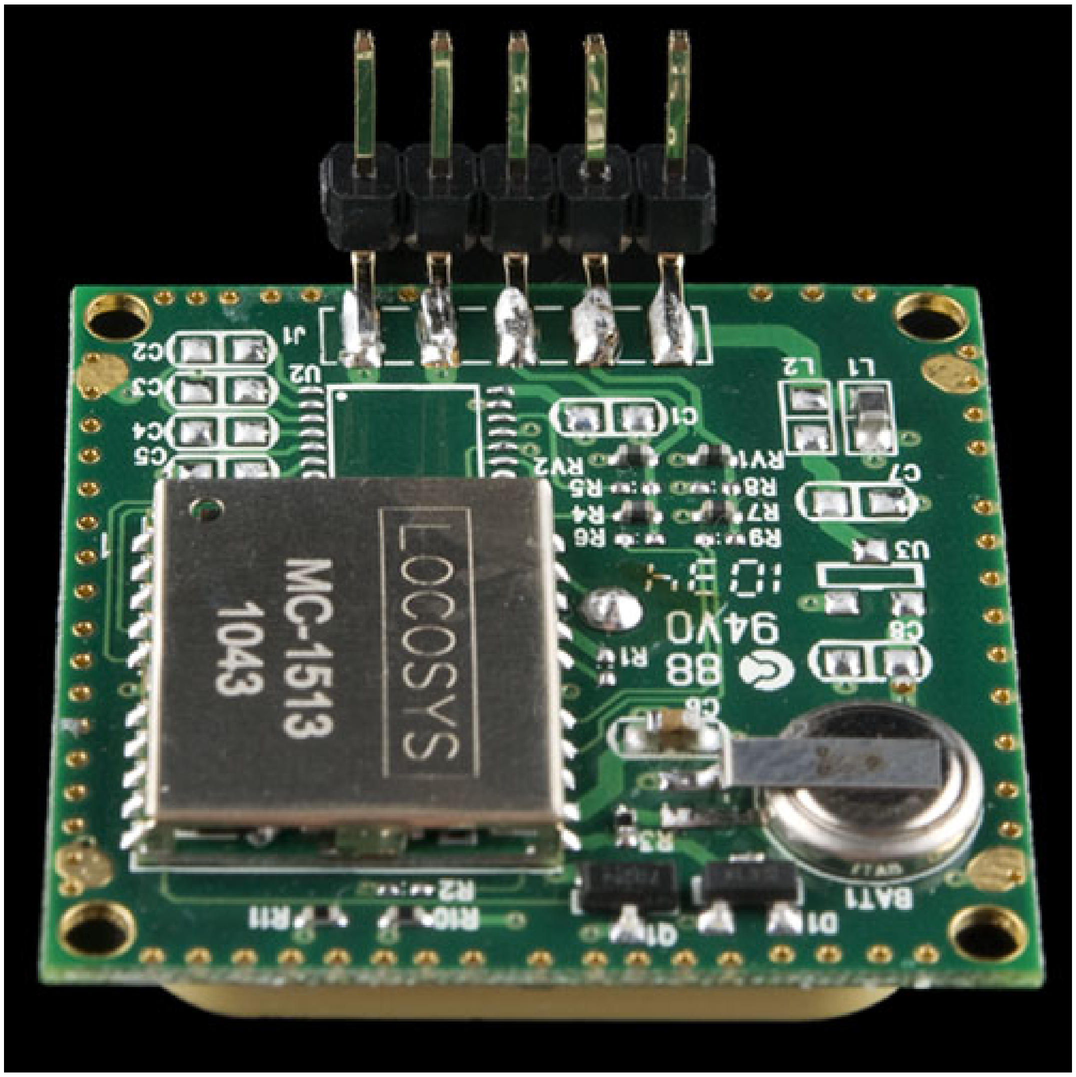
Next remove the temporary adhesive and ensure the pins are still aligned correctly on the pads—if they're not, apply heat to the already soldered connector and gently move the connector into alignment.

Then solder the other outside pin into place like this:



Finally, solder the remaining three pins to their pads. With the two outside pins in place the remaining pins should stay in alignment with the pads.

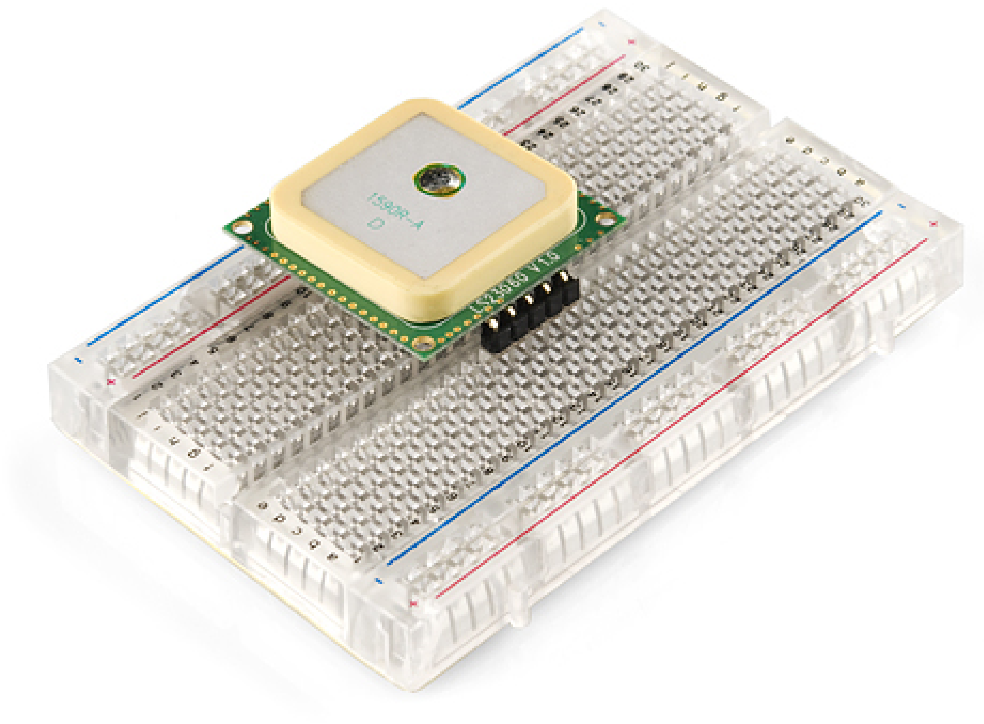
When you've finished your module should look something like this:



You're done! Read on to the next section to find out how to wire your GPS module to an Arduino for a quick test.

Completed

Now that your GPS module has a right-angle connector you can plug it directly into a breadboard like this:



In order to retrieve data from the GPS module we will connect it to an Arduino. The LS20031 requires 3.3V for power and according to the product pages requires 41mA of current so we can use the Arduino's 3.3V pin for power as that can supply up to 50mA. The only complication with using the module connected to a standard 5V Arduino is that the module can only communicate with a maximum of 3V on its receive (RX) and transmit (TX) connections. For our test connections we will not connect the RX pin of the module (as that would require voltage level conversion), only the TX pin—which will transmit the data to a pin on the Arduino. (The Arduino will recognise the maximum of 3V as a "high" value so it can still understand the data being transmitted by the GPS module.)

Quick test

To perform a quick test, first upload the following sketch to your Arduino IDE (Pre-1.0):

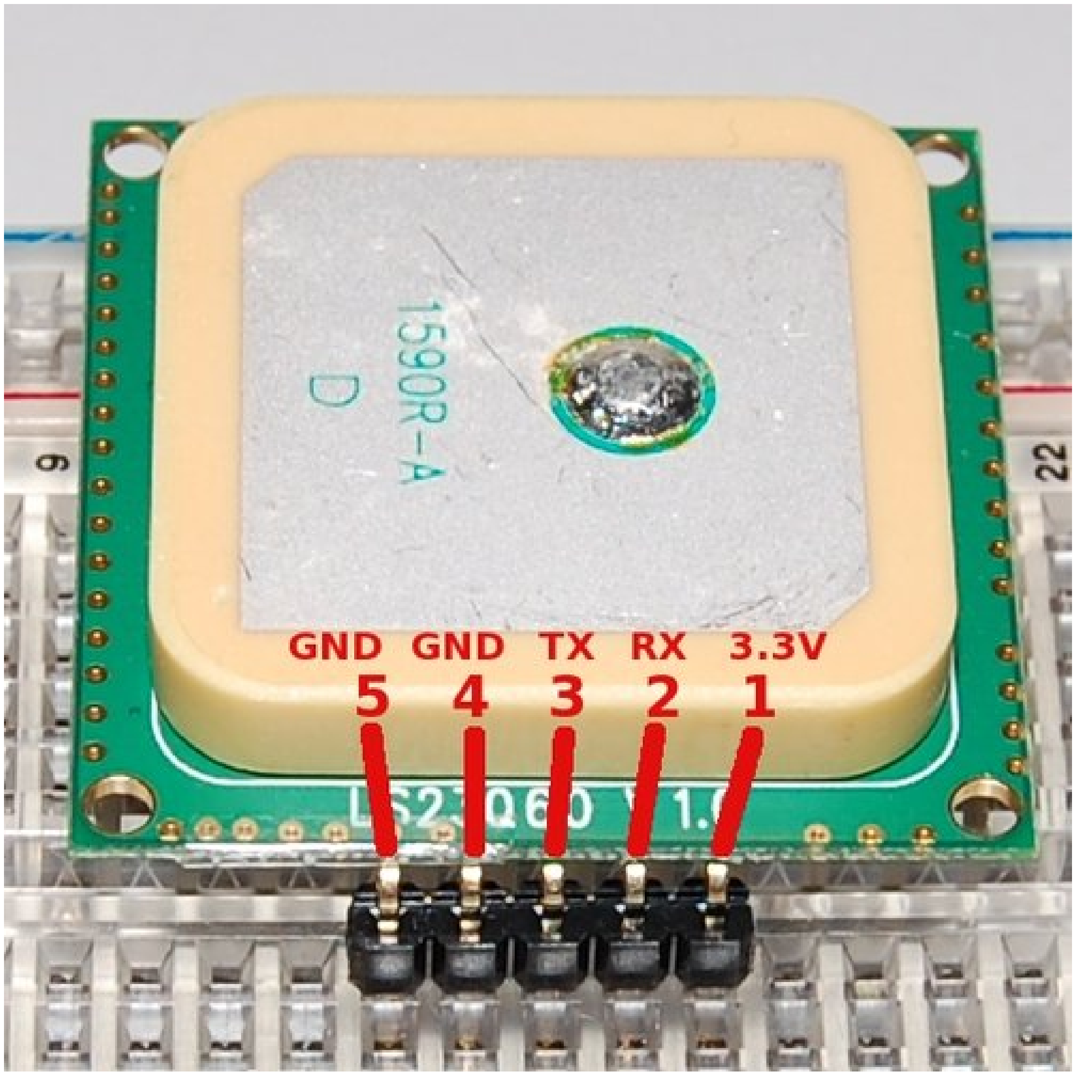
```
void setup() {
  Serial.begin(57600);
}

void loop() {
  if (Serial.available()) {
    Serial.print(Serial.read(), BYTE);
  }
}
```

If you're not sure what version of the Arduino IDE you're using, or if you're on 1.0+, you can use:

```
void setup() {
  Serial.begin(57600);
}
void loop() {
  if (Serial.available()) {
    #if ARDUINO >= 100 //For Arduino v1.0+
    Serial.write(Serial.read());
    #else //For Arduino v0023 or earlier
    Serial.print(Serial.read(), BYTE);
    #endif
   }
}
```

Then wire the GPS module and Arduino together like this:

- GPS Pin 5 (**left-most when viewed from above**) : No connection (or ground)
- GPS Pin 4 : to Arduino ground (GND) pin
- GPS Pin 3 : to Arduino pin 0
- GPS Pin 2 : No connection
- GPS Pin 1 (**right-most when viewed from above**) : to Arduino 3.3V pin



(We actually cheat a little here because we end up with both the computer and GPS connected to the serial connection of the Arduino—but because we only ever receive data from the GPS module and only ever send data to the computer we can have both connected at the same time for our quick test. This is why we need to upload the sketch before we wire the connections up. To find out how to wire the devices properly see the next section.)

Now connect your Arduino to your computer, open the Arduino IDE Serial Monitor and ensure it is set to communicate at 57600 baud. You should then see information being written to the Serial Monitor like this:

```
$GPGGA,105317.709,8960.0000,N,00000.0000,E,0,0,,137.0,M,13.0,M,,*4C
$GPGLL,8960.0000,N,00000.0000,E,105317.709,V,N*49
$GPGSA,A,1,,,,,,,,,,,,,,,*1E
$GPGSV,1,1,00*79
$GPRMC,105317.709,V,8960.0000,N,00000.0000,E,0.00,0.00,010610,,,N*78
$GPVTG,0.00,T,,M,0.00,N,0.00,K,N*32
```

The information is displayed as NMEA sentences which the GPS module uses to communicate position information.

If nothing is displayed you'll need to double check your connections and soldering.

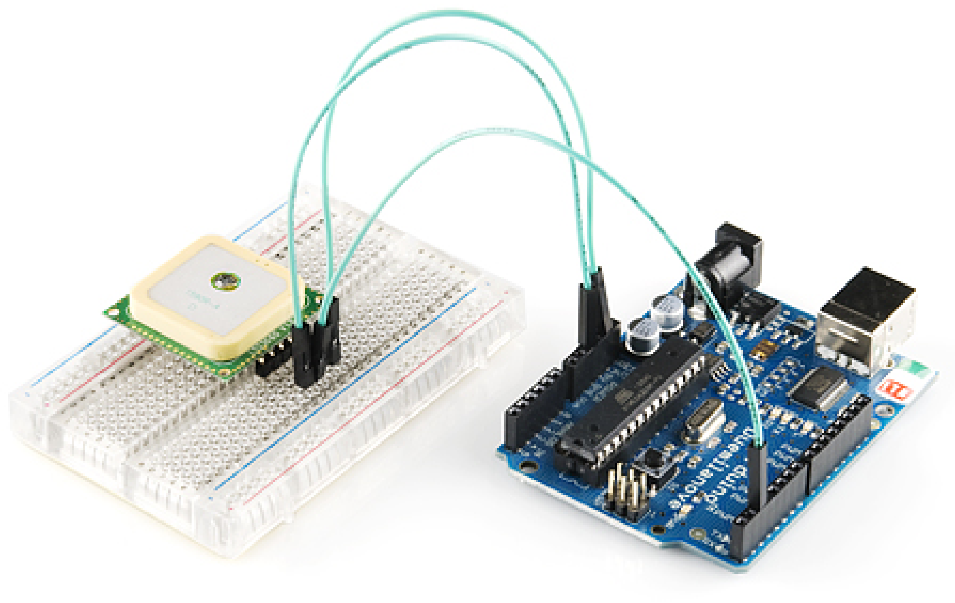
Using the TinyGPS library

Fortunately we don't have to interpret the information the GPS sends out ourselves as there's a really helpful library TinyGPS that will do the hard work for us.

Install the TinyGPS library and then change the wiring so GPS pin 3 now connects to Arduino pin 2.

Here is a list and photograph of the required connections:

- GPS Pin 5 (**left-most when viewed from above**) : No connection (or ground)
- GPS Pin 4 : to Arduino ground (GND) pin
- GPS Pin 3 : to Arduino pin 2
- GPS Pin 2 : No connection
- GPS Pin 1 (**right-most when viewed from above**) : to Arduino 3.3V pin



Then open the example sketch `test_with_gps_device` included with TinyGPS library from `File > Examples > TinyGPS > Examples > test_with_gps_device`.

Because the default communication speed of this GPS module is faster than some other modules you will need to replace the following line in the sketch:

```
nss.begin(4800);
```

with this line:

```
nss.begin(57600);
```

Upload the sketch, then open the Serial Monitor and this time make sure it is set to communicate at 115200 baud.

At first the Serial Monitor will display a message like this:

```
Testing TinyGPS library v. 9
by Mikal Hart

Sizeof(gpsobject) = 103
```

After a while you should see position reports like the following:

```
Acquired Data
-------------
Lat/Long(10^-5 deg): -4223579, 17344651 Fix age: 14ms.
Lat/Long(float): -42.23579, 173.44651 Fix age: 22ms.
Date(ddmmyy): 0 Time(hhmmsscc): 11210560 Fix age: 35ms.
Date: 0/0/2000  Time: 11:21:5.60  Fix age: 40ms.
Alt(cm): 13870 Course(10^-2 deg): 999999999
 Speed(10^-2 knots): 999999999
Alt(float): 138.70 Course(float): 10000000.00
Speed(knots): 10000000.00 (mph): 11507795.00 (mps): 5144444.00
 (kmph): 18520000.00
Stats: characters: 156236 sentences: 10 failed checksum: 35
-------------
```

The first time you connect the module it may take a few minutes to determine an initial location. If you still don't have a position after a few minutes try re-locating the GPS module so it has a clearer view of the sky and ensure the ceramic antenna is facing the sky.

Once the GPS module has acquired a position the red light on it will blink.

Well done, now you know where you are! Good luck getting home...

Have a suggestion for how we can improve this assembly guide? Steps missed? Instructions unclear? Please let us know. You can leave a comment below or email us spark@sparkfun.com. Also let us know if this is the most awesome assembly guide you've ever encountered and we'll stop trying to improve it.

Comments 48 comments

Log in or register to post comments.

smremde / about 5 years ago / ★ 6

To the above commenter, editor and anyone else who reads this:
I had the same trouble, then I noticed the large number of failed checksums. Long story short, the default buffer in the NewSoftSerial library is too short for this device (probably because of the device's 5hz update and fast baud rate.)
Open up \libraries\NewSoftSerial\NewSoftSerial.h and change

define _NewSS_MAX_RX_BUFF 64 // RX buffer size
to

define _NewSS_MAX_RX_BUFF 128 // RX buffer size
to fix this!
Hope this helps :)

Member #508928 / about a year ago / ★ 1

This confused me for a long time but on the newer Arduinos I had to go into the C drive to find the SerialSoftware.h file and changed that to 256. I had originally downloaded the NewSoftSerial software and was changing that and it never worked.

Mark Fickett / about a year ago * / ★ 1

On a Mac with Arduino 1.0.5, note that SoftwareSerial is inside Ardiuno.app (/Applications/Arduino.app/Contents/Resources/Java/libraries/SoftwareSerial/SoftwareSerial.h) and the constant is just _SS_MAX_RX_BUFF . (NewSoftSerial is no longer required with Arduino 1+.) I ended up setting it to 256, since a GGA + RMC sentence is about 140 characters.

Blacklab1 / about 4 years ago / ★ 1

That worked the first time. And boy did it come through clean- no hickups.
I wonder if Nate @ Sparkfun updated his code?
Thanks Smremde.
-Blacklab

TimDC / about 2 years ago / ★ 2

If you're having trouble hooking up this module and you're using right-angle solder pins, be sure to check your solder joints very carefully for a good connection. It's very easy for solder to "blob" on top of the pin instead of flowing properly b/t pin and pad. I scratched my head for several days trying to figure out why my module wasn't drawing any current. I was convinced the module was bad, but then I carefully reflowed the solder on the VCC and GND pins and it started up like a dream.

rubot / about 5 years ago / ★ 2

AArrg, this works great on my Uno but i can't get it to work on my Pro Mini. Any reason TinyGPS won't work with an Arduino Pro Mini 3.3v?

M-Short / about 4 years ago / ★ 1

It might be because the Pro Mini 3.3V runs at 8MHz, this can cause problems with some baud rates and in general just runs slower.

Sora62896 / about 4 years ago / ★ 1

Mine also works on my uno, but I can’t get it to work on my mega2560! Is this the same issue?

M-Short / about 4 years ago / ★ 1

The Mega2560 runs at 16MHz so it should work fine. But the Mega2560 does not have the proper interupts to do software serial on those lines which might be your problem. Check out the software serial library page or email us in tech support if you have any other questions.

Sora62896 / about 3 years ago / ★ 1

figured it out–needed to change “SoftwareSerial nss(3, 4);” to “# define Serial1”

Joss / about 4 years ago / ★ 1

Same issue here - very frustating! Did you ever resolve? It seems all ok on the hardware RX but not via Software Serial! I’ve tried all the updated Arduino 1.0 and Tiny GPS v12. Everything is fine on the Uno but not with the 3.3v Pro Mini. The data seems to come through but it is all garbled - possibly a tiing issue in Software serial?

Victer0409 / about a year ago * / ★ 1

In the „test_with_gps_device” code, I only get an increasing number of Chars. The rest of the fields are filled with stars (*), except the Sentences and Checksum, these are both 0. I’ve already increased my _SS_MAX_RX_BUFF to 256. And I have a fix (blinking red led on the ls20031). Does anyone have any idea what the cause of this problem could be? SOLVED! had the wrong baud rate

Member #618416 / about a year ago / ★ 1

I am running this gps module to a arduino uno and i just added the gps library and changed the pins to pin 3 and i am getting this: PLEASE HELP!! Testing TinyGPS library v. 13 by Mikal Hart

Sats HDOP Latitude Longitude Fix Date Time Date Alt Course Speed Card Distance Course Card Chars Sentences Checksum

```
      (deg)     (deg)      Age                          Age   (m)     --- from GPS ----
---- to London  ----   RX     RX         Fail
```

---

0 0 0
0 0 0
0 0 0

0 0 0
0 0 0
0 0 0
0 0 0
0 0 0
0 0 0
0 0 0
0 0 0
0 0 0
0 0 0
0 0 0
*** 0 0 0

Mark Fickett / about a year ago * / ★ 1

If your unit can see 4+ satellites but never gets a fix/lock, send a FULL_COLD_RESTART.

I was stuck because my unit would report that it could see up to 9 satellites (the “09” in the below GSV sentences):

```
$GPGSV,3,1,09,05,,,39,29,,,31,26,,,42,02,,,38*7F
$GPGSV,3,2,09,21,,,20,15,,,18,30,,,29,10,,,24*72
$GPGSV,3,3,09,07,,,25*70
```

but it never started producing position data / blinking the red LED. SparkFun support advised me to send a FULL_COLD_RESTART:

```
Serial.println("$PMTK104*37"); // FULL COLD RESTART
```

Details are in https://www.sparkfun.com/datasheets/GPS/Modules/PMTK_Protocol.pdf . This resets the GPS’s almanac, which very rarely comes with bad data or something. Note that unlike the example code where the GPS’s 3.3v output can be fed into a 5v Arduino (I was using an Uno), to send data to the GPS you’ll have to have a 3.3v Arduino or a logic level converter (like BOB-12009).

Also, TinyGPS only uses GGA and RMC sentences, so I configured my unit to only send those, and only once every 5 fixes (that is, 1Hz):

```
 Serial.println("$PMTK314,0,5,0,5,0,0,0,0,0,0,0,0,0,0,0,0,0*28");
```

Member #510425 / about 2 years ago / ★ 1

This is very confusing because the only thing that i am getting from the test_with_gps_device is this *** 711 0 0 . The only things I changed are the ss.begin(57600) and the SoftwareSerial ss(1, 0); instead of SoftwareSerial ss(4, 3);. Help please? thanks :)

Ezu / about 2 years ago / ★ 1

In this tutorial I find what I need to start interfacing the gps sensor with a SBC. And because I want to help many more hobbyists to start building robots, I share this tutorial on my post http://www.intorobotics.com/gps-sensors-tutorials-resources/. Thank you!

Member #459842 / about 3 years ago / ★ 1

sir pls give me the information to connect ls20031 to pc

Member #336767 / about 3 years ago * / ★ 1

I used TinyGPS v12 example code “test_with_my_gps_device” with arduino mega board. ls20031 was locked and gives valid data.

(I checked it using this code

void setup() { Serial.begin(57600); } void loop() { if (Serial.available()) { #if ARDUINO >= 100 //For Arduino v1.0+ Serial.write(Serial.read()); #else //For Arduino v0023 or earlier Serial.print(Serial.read(), BYTE); #endif } }) But “test_with_my_gps_device” code shows ******* as output.

“test_with_my_gps_device” output

Testing TinyGPS library v. 12 by Mikal Hart

Sizeof(gpsobject) = 115

Sats HDOP Latitude Longitude Fix Date Time Date Alt Course Speed Card Distance Course Card Chars Sentences Checksum

```
      (deg)    (deg)     Age                           Age  (m)     --- from GPS ----
---- to London  ----  RX    RX        Fail
```

---

0 0.00 **0 0 0**
0 0.00 **0 0 0**
0 0.00 **0 0 0**
0 0.00 **0 0 0**
0 0.00 **0 0 0**
0 0.00 **0 0 0**
0 0.00 ** 0 0 0
Need a big help Thank you.

> Blacklab1 / about 3 years ago * / ★ 2
>
> Make sure your wiring is correct, and check Mikal Hart web page on what people did to make sure it works. I remember having this problem. < EDIT > Make sure your TX on your LS20031 is going to the correct Pin on your Arduino. AND make sure your code reflects that >> / *EXAMPLE: The circuit: * GPS TX Pin 3 : to Arduino pin 3- I know Sparkfun show this on Pin 2, but try it.*
> ** GPS RX Pin 2 : **No connection** unless using a Logic Level Converter * **See the link on how to hook things up bellow. * Just make sure your wiring reflects how you have set up your code. /* #define RXPIN 3 // From the GPS to the Arduino #define TXPIN 2 SoftwareSerial nss(RXPIN, TXPIN); void

loop() { while (nss.available()) { int c = nss.read(); if (gps.encode©) { // process new gps info here } } } Yeah, I know the #define adds a little more typing, but its a lot easier to read, and less likely to make mistakes. Also, using #define does not eat up memory in your program.

IF you have TX on the LS20031 going to TX on the Arduino then TinyGPS will give you a bunch of zeros and a headache.

SIDE NOTE >> If you are skipping SoftwareSerial meaning your feeding directly to PIN 0 on the Arduino from you LS20031 (Pin 3) you need to make sure you are reflecting that in your code, meaning your **NOT** using SoftwareSerial (I am sorry I don't have an example of that right now). And YES, you can directly feed your Arduino using the on board hardwired UART to check TinyGPS. At this time, I personally find this a lot easier than trying to use Software Serial.

If you are using SoftwareSerial then PIN 3 on the Arduino needs to be hooked up to Pin 3 on the LS20031. Also, You need to get the data sheet for LS20031 so you can become more familiar with the chip and how to read the NMEA sentences. Also it won't hurt you to print out TinyGPS ver 12 and try to read what it does. Reason I am pointing this out, if you look closely at TinyGPS, it only uses two of the sentences the LS20031 spits out - $GPGGA and $GPRMC. As it has been pointed out here, that means you can turn off everything else.

IF you want an example about how to go about turning off things you need take a look at LadyAda web example here. BTW- this example requires you using SoftwareSerial.h. No hardwired UART.

YES, I know her code is not for the LS20031, but if you read the data sheet of the LS20031 then you can fill in the correct code to do the job.

Which means you can turn all the other sentences off, but then you're going to need a Voltage Level Converter, so you can talk to the (3 volt) LS20031 from the (5 Volt) Arduino. And **YES**, you can use either the I2C voltage level converter, or Logic Level Converter. I have a I2C converter working, and yes you can tell the difference with a scope in both directions. I really like this version of converter because it's really simple and direct- and you don't have to turn over the part to figure how to wire it.

>> Click for Picture of how things should be hooked up using a Logic Level Converter

You also might want to read Wayneholder's blog about Navigating with GPS. If your GPS driven car is jumping all over the place he has some great ideas how to fix it.

Blacklab1 / about 3 years ago * / ★ 1

Also, if you look above it seems Tritaris has got a track on what you need to change. so something like this >>

```
/* LS23001 EXAMPLE:
The circuit:
• Pin assignments for the LS20031 GPS:( http://www.sparkfun.com/tutorials/
176 )
• GPS GND Pin 5 (left-most when viewed from above) : No connection (or gro
und)
• GPS GND Pin 4 : to Arduino ground (GND) pin
• GPS TX  Pin 3 --> Logic Level Converter --> to Arduino RX Pin 3
• GPS RX  Pin 2 < --Logic Level Converter < -- from Arduino TX Pin 2
• GPS 3.3 Pin 1 (right-most when viewed from above) : to Arduino 3.3V pin
```

See Picture how to hook up Logic Level Converter * Just make sure your wiring reflects how you have set up your code. */

```
#define RXPIN 3 // From the GPS to the Arduino
#define TXPIN 2
#if (ARDUINO >= 100)
#include
SoftSerial mySerial(RXPIN, TXPIN);
#else
// If you're using Arduino IDE v23 or earlier, you'll
// need to install NewSoftSerial
#include
NewSoftSerial mySerial(RXPIN, TXPIN);
#endif
void setup()
{
Serial.begin(115200);
mySerial.begin(57600); // 57600 NMEA is the default baud rate for LS20031

//the following Turned off all NEMA sentences except GGA and RMC
mySerial.println($PMTK314,0,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,028\r\n);

mySerial.println($PMTK220,1002F\r\n); // Set the update rate to 10Hz outpu
t

mySerial.println($PMTK301,22E\r\n); // Turned on WAAS
}

 void loop()                     // run over and over again
 {
 if (mySerial.available()){
   Serial.print((char)mySerial.read());
   }
 if (Serial.available()) {
   mySerial.print((char)Serial.read());
   }
}
```

Blacklab1 / about 3 years ago * / ★ 1

Now if you want to go and change one of the commands your going to need to have a check sum at the end of your command. Lada Ada has alink for it >> Awesome. But if you can convert this so the Ardunio IDE can run it, here's an idea from wikipedia >>

```
#include < stdio.h >
#include < Strings.h >

int checksum(char *s) {
 int c = 0;
 while(*s) c ^= *s++;
 return c;
 }

int main()
{
char mystring[]= "GPRMC,092751.000,A,5321.6802,N,00630.3371,W,0.06,31.
66,280511,,,A";

printf("String: %s\nChecksum: 0x%02X\n", mystring, checksum(mystrin
g));
return 0;
}
```

Jason.C / about 4 years ago / ★ 1

What material is being used in the Assembly photos to hold right right-angle header to the GPS while doing the initial soldering?

Blacklab1 / about 3 years ago * / ★ 1

you mean the white stuff in Picture 4 and 5 ? I think it's clay. Just make sure the type you get is not plastic, because when you heat it, plastic melts

Sora62896 / about 4 years ago / ★ 1

Just a warning for anyone who tries soldering to the bare pads, try avoiding using individual right angle headers rather than the bundle of five. I order a bundle of six, and when I tried breaking one off, another came with it. The single one which was soldered to the second ground pin broke right off of the board, solder and all…..leaving me with one less ground–but still functional!!! Please be careful!!

Blacklab1 / about 4 years ago * / ★ 1

… This web page will not let me show the code I found that was being a problem child. And why you might be having a problem with TinyGPS and the 1.0 IDE … Hopefully Sparkfun will catch it…

Blacklab1 / about 4 years ago * / ★ 1

IF your going to try the Quick test GPS code your going to need to change it so it looks like this to work with the Arduino 1.0 IDE >>

void setup() {

Serial.begin(57600);

}

void loop() {

if (Serial.available()) {

```
Serial.write(char(Serial.read()));
```

//it was pointed out that read() needs to be type casted to char()

}

}

egrav / about 4 years ago / ★ 1

I just bought one of these, and the comments were quite useful to get the unit to work.
A word of warning - After soldering the right angle header to the GPS board, be very careful when removing GPS plugged into breadboard. The breadboard tends to grip the pins tightly, and I actually tore the traces off of the GPS board, leaving the pins stuck in the breadboard. I was able to repair the board, and then I added a 2 x 2 perfboard under the GPS, so I can pull the unit out by the perfboard.

Tritaris / about 5 years ago * / ★ 1

I was having the same problems with course, speed, time and date showing invalid data. Here is what I did to get it up and running:
In test_with_gps_device.pde:
nss.begin(4800); to nss.begin(57600);
In NewSoftSerial.h:
NewSS_MAX_RX_BUFF 64 to NewSS_MAX_RX_BUFF 256
Used Indigo Terminal Emulator to send the following commands to the GPS module.
Turned off all NEMA sentences except GGA and RMC
$PMTK314,0,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,028*\r\n*
Set the update rate to 10Hz output
*$PMTK220,100*2F\r\n
Turned on WAAS
*$PMTK301,2*2E\r\n
*Optional
Websites used:
http://diydrones.com/profiles/blogs/using-the-5hz-locosys-gps-with
http://dallasmakerspace.org/wiki/LS20031_GPS
NMEA checksum calc

Sora62896 / about 4 years ago / ★ 1

I don’t have a NewSoftSerial library or anything in the arduino files—is softserial the same file?

M-Short / about 4 years ago / ★ 2

As of Arduino 1.0 NewSoftSerial was made an official library in the Arduino IDE which is just called SoftwareSerial. The new TinyGPS library should take that into account and not require NewSoftSerial.

V3_XD / about 4 years ago / ★ 1

Hello. What did you use to send the commands to the GPS? Did you buy the FTDI cable or did you send it through the Arduino?

Sora62896 / about 4 years ago / ★ 1

Connecting it through the Arduino is much easier especially with the tutorials provided by sparkfun!!

josefodunga / about 5 years ago * / ★ 1

I’m also having the same problem as Federico.
Using the test sketch I always get the same speeed, the date won’t update (it stays on 0/0/2000), and I get and aditional checksum added on every serial print.
I got less checksum errors when doing reads more often as somebody suggested but they are still there.
Anybody know how to resolve this?
Finally I got a solution. Updating the newsoftserial library value of NewSSMAXRXBUFF to 256 I finally got it to work.

Sora62896 / about 4 years ago * / ★ 1

where did you find “NewSSMAXRXBUFF”?

How exactly did you get to this? I’m looking in the libraries…..

Sora62896 / about 4 years ago / ★ 1

never mind—not “New”– just SSMaxRxBuff (caps may be off) in softserial–but still no date!

Member #210910 / about 5 years ago / ★ 1

I am curious why a standard RS232 port, since they all have UARTs these days, and minicom would not work? Saves having to get an Arduino board. Not going to be able to do anything useful with it, but it should be a good check for signs of life anyway.

Member #159801 / about 5 years ago * / ★ 1

I have used the above example and I have gotten it to work fully. I also used parts of the example from http://diydrones.com/profiles/blogs/using-the-5hz-locosys-gps-with. I used the commands from DIY to have GPS only print out GPGGA and GPRMC data at 5hz. I also decreased baud of the gps to 14400. I get full data dumps at 5hz when I increase the update repeat time to .15 secs (while (millis() - start < 150)). Seems to be less failed checksums at higher rates.

Fede IK0ADR / about 5 years ago / ★ 1

Hello.
Is there anyone out there able to receive a correct date from this?

> Date: 0/0/2000
> Thanks,
> –Federico

Sora62896 / about 4 years ago / ★ 1

I have had the same issue–the date is incorrect and there is no speed— I don't even know about the heading–I'm using a separate compass for that.

MichaelShimniok / about 5 years ago / ★ 1

Well… I connected per above and the datasheet, am getting NMEA data but the module isn't getting any fix, after many minutes (30) in various spots including outside. No red blinky light (only comes on at powerup)
I've sent email to SFE tech support and customer support twice now… no response… hello? Anyone out there? :)
I think the unit is defective. I posted on SFE's forum, no response in a few days.
Posted on Pololu's forum and got a response within minutes with a helpful link to a forum thread talking about solder connections between the metal shield and pcb.
So, maybe I have a defective unit. If I can ever get in touch with SFE maybe I can find out what else to try. :)

David Varney / about 5 years ago / ★ 1

I too am having the same issue as Smolders. The quick test is working but I get no results with the TinyGPS Library. I get no errors when I compile and I've made sure my connections are 100%. Any help would be greatly appreciated.

David Varney / about 5 years ago / ★ 1

After waiting for what seems like forever the setup actually worked for the TinyGPS example. Could somebody point me towards some better examples with this TinyGPS library and/or a method of which works quicker?

Smolders / about 5 years ago / ★ 1

I followed the steps and the Quick Test is working. But when I use the TinyGPS library, I cant get information out of my GPS.. Even when the red LED is blinking which means the GPS is fixed.
Any help?

> Blacklab1 / about 4 years ago / ★ 1
>
> Have you tired the fix that smremde showed? It worked for me

Pog / about 5 years ago / ★ 1

I too found that the course and speed data were missing (set to default). As an experiment, I tried viewing the data as it came in through the NewSoftSerial ports using this:

include

NewSoftSerial nss(2, 3);
void setup()
{
Serial.begin(115200);
nss.begin(57600);
}
void loop() {
if (nss.available()) {
Serial.print(nss.read(), BYTE);
}
}
Even with the buffer NewSS_MAX_RX_BUFF cranked up to 256 (and I tried higher), you can still see that data is being lost when you view the serial monitor (vs. the data you see when you use the QuickTest sketch and pin0) Clearly things are better when you use the hardware UART. I'm thinking that NewSoftSerial just isn't fast enough? Though maybe someone else will have more detailed knowledge and can diagnose what is really going on…

sfd / about 5 years ago / ★ 1

I was also having the issue where the speed and date are not updated.
The raw NMEA messages output from the Quick test sketch are fine, so the problem is either in TinyGPS or NewSoftSerial.
With the test_with_gps_device sketch it looks like a lot of characters are getting dropped, so no GPRMC messages are being parsed, and so the speed and date are never updated. The displayed numbers are the 'invalid' indicators. This is probably because of the high baud rate.
The suggestion from smremde to increase the NewSoftSerial buffer size helped, but nss.overflow() still returns true, indicating that data from the GPS is still being dropped.
I changed test_with_gps_device to use the h/w serial port instead (like the Quick test), and found that worked better. The date/speed are reported, and it does not seem to be dropping data. The downside is you are sharing the h/w serial port with the PC so you have to disconnect the GPS to upload sketches.
Hope this helps other with the same issue.

RobertC. / about 5 years ago / ★ 1

Make sure you are using the newest versions of the libraries, as well as the newest Arduino IDE. If you are still having trouble, email techsupport@sparkfun.com. Thanks.

Blanco / about 6 years ago / ★ 1

How do I get the course and speeds to work? In the above example I think they are set to their default values when there is no valid reading.
I was easily able to parse the NMEA data using Processing, but I want to do the same thing on my Arduino.