

[Home](#) [Blog](#) [Reviews](#) [Register](#)

# Laptop GPS World

Mapping and GPS Navigation for Computers, Laptops, PC, Windows, Mac, Linux, Tablet, Netbook, iPad, UMPC, CarPC

 User Name

 Password



[Laptop GPS World](#) ► [Laptop GPS Hardware](#) ► DIY GPS module using the Locosys LS20031

## DIY GPS module using the Locosys LS20031



Thread Tools ▾

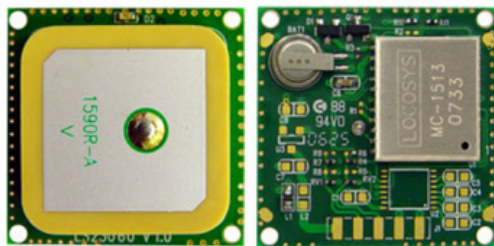
Jul 11, 2010, 09:53 PM

#1

### L6E

#### DIY GPS module using the Locosys LS20031

So why would you want to fiddle with your own OEM GPS module? Well it could be that the functions you're looking for are not available in a packaged module (for instance the LS20031 can be configured so different NMEA sentences are repeated at different intervals). It could be that you want to know exactly how your GPS module acts under certain circumstances (many packaged modules are programmed with what might have seemed like handy personality traits at the time). Maybe you need a unique interface connection or maybe you have a custom application in mind like controlling an R/C helicopter or building a remote tracking device. Maybe you just want to play with and learn about electronics.



The LS20031 is a high performance GPS receiver and antenna. I'd list all the cool things about this module but by the time I do you might as well read the datasheet.

I should note here that if you are planning to use this module exclusively on a computer RS232 serial port there is the LS20032 which already has RS232 compatible voltages at the RX and TX pins.

There is also a USB version, the LS20030. I'm not familiar with this one so you're on your own as far as USB drivers, etc.

**TIP:** Pay attention to the power pin voltage requirements on these different versions. They differ to suite their particular application.

There is also an ultra tiny, no bells and whistles version of this module called the LS20033.

From the LS200XX series I chose the LS20031 version for the fawn tracking project posted here: [GPS Tracking of Deer Fawns Directly to Your Laptop! \(Locosys LS20031\)](#) because it had unmodified TX/RX voltages that I could convert to whatever application I desired and also because it had a full size antenna (I don't expect the fawns to be going out of their ways to position the GPS toward clear skies on queue). I'm also quite happy about the 4 mounting holes built into this version.

You'll pay \$60-\$70 for this module. That's a little more than the typical \$45 for a module of similar size BUT... the LS20031 has higher performance and more bells and whistles than comparable units. Noteworthy here is it's 5 times a second update rate, fast time to fix, and its flexible configuration. It is also splayed out on an extremely high quality printed circuit board of likes you rarely see outside the aerospace industry. This speaks volumes for the units overall workmanship. I have designed dozens if not hundreds of PCBs and there is something je ne sais quoi about this little guy.

The datasheet for these modules can be found here: [http://elmicro.com/files/sparkfun/ls...asheet\\_v10.pdf](http://elmicro.com/files/sparkfun/ls...asheet_v10.pdf)

The datasheet is ridiculously anorexic. Included in what it won't tell you is the upper communication baud rate that the LS200 series can be configured to. What it will tell you is that it is shipped with a default bps of 9600. This is completely false. At least in my case it was shipped set to 57600 bps. That is wayyyyyyy fast for low power consumption applications and many serial port applications. Snooping around I've found that others have had it shipped at 4800 bps. This is important because even if you are planning on changing the baud rate you need to find out what it is originally set to in order to communicate with it to make changes. Fortunately for me I have an oscilloscope and was able to read the output and measure the bit rate. For those of you who don't you'll have to use another fairly simple trick which I'll get into in a bit.

Another thing the data sheet won't tell you is the LS20031's weight, approximately 12grams.

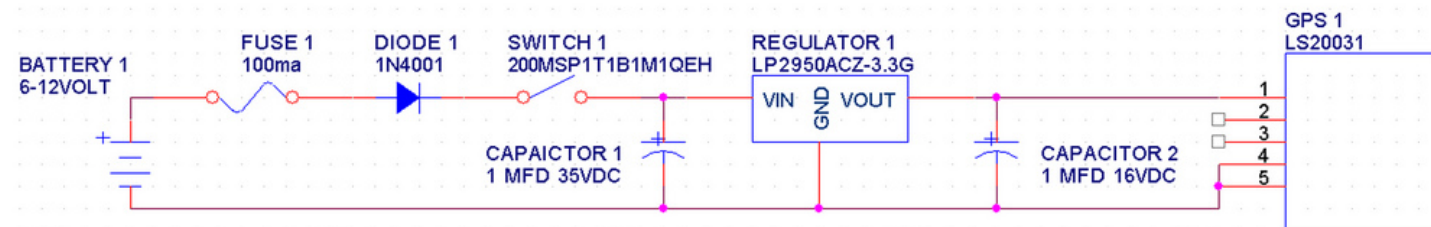
**POWERING THE LS20031 MODULE**

The first thing you will want to do with the LS20031 is power it up. Here is the pinout:

**PIN1 VCC (POWER IN, +3.3VDC)**  
**PIN2 RX (DATA INPUT TO GPS)**  
**PIN3 TX (DATA OUT FROM GPS)**  
**PIN4 GND (POWER & DATA GROUND)**  
**PIN5 GND (POWER & DATA GROUND)**



It requires 3-4.2 volts dc (designed for the 3.3 volt industry standard). It requires less than 40milliamps so just about any 3.3 volt 3pin fixed regulator will do. Unfortunately 5 volts is too high for these modules so using the 78xx5 series regulators that all electronics hobbyists have in their junk collection won't do...don't even think about it! Regulators are available over night at Digi-Key and Newark in One for less than a dollar. If you are using the module for an automotive application make sure you use a regulator (and filter capacitors) that exceed the input voltage which a vehicle will supply. Many of these regulators are rated for 18 Volts and up which is perfect for this. If you are building a hand held unit or something else that requires low power consumption I would recommend a switching regulator such as the Texas Instruments PTH0808. It is small, has plenty of overhead current for running other parts of the circuit (2.25Amps) and costs around ten dollars. I would also recommend implementing a fuse and rectifier at the input of your power supply circuit (to avoid accidental reverse voltage). Here is how I would do it:

**Bill Of Materials**

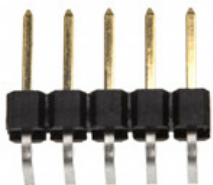
Qty Reference Part Suggested

Qty	Reference	Part Suggested
1	BATTERY 1	6-12VOLT (OR POWER SUPPLY)
1	CAPACITOR 2	1 MFD 16VDC 718-1209-ND
1	CAPACITOR 1	1 MFD 35VDC 718-1204-ND
1	DIODE 1	1N4001 1N4001FSTR-ND
1	FUSE 1	100ma F2356-ND
2	CLIP 1, CLIP 2	FUSE CLIP F063-ND
1	GPS 1	LS20031 LS20031
1	REGULATOR 1	LP2950ACZ-3.3G LP2950ACZ-3.3GOS-ND
1	SWITCH 1	200MSP1T1B1M1QE EG2446-ND

Apologies for the squished together BOM. The forum doesn't allow multiple spaces or tabs.

**TIP:** I really like the Littelfuse 01000056H (Digi-Key part# F063-ND) clips for circuit board and perfboard mounting of fuses. They hold 25mm fuses such as the 0216.100HXP (Digi-Key part# F2356-ND). Requires two clips per fuse.

**TIP:** The LS20031 connection pads have .100 spacing so for easy use with a prototype bread-board a 90 degree header such as the 961105-5604-AR (Digi-Key Part# 3M9470-ND), or 87233-5 (Digi-Key Part# A28771-ND) can be soldered on.



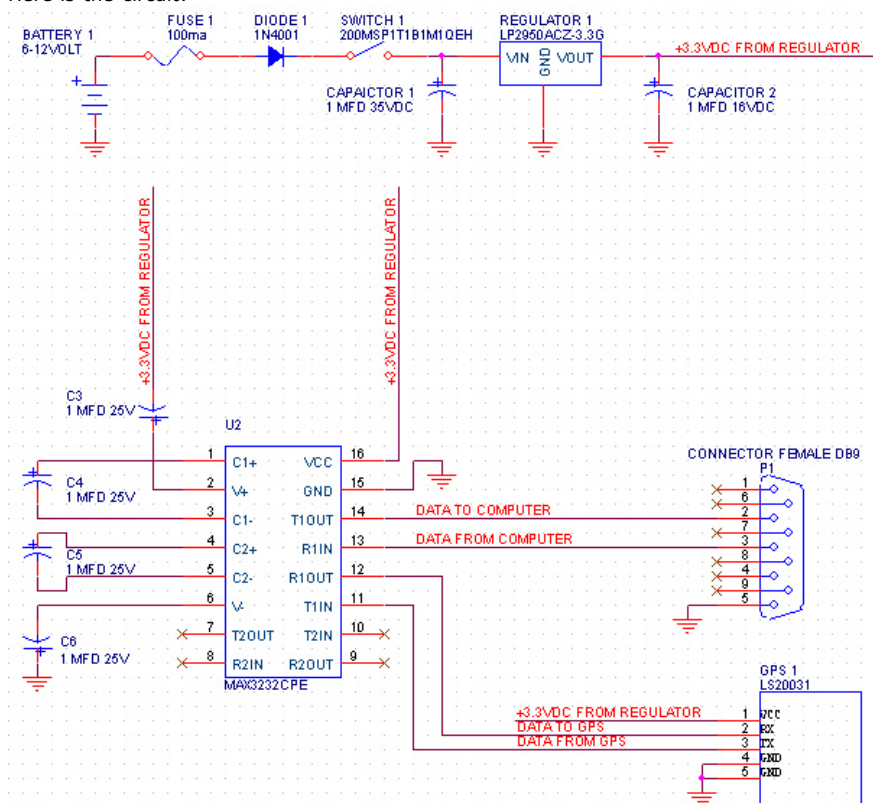
That's it for powering the module up. Once power is applied the LS20031 will start transmitting NMEA data out of the TX pin 3. If there are satellites in view, the LED on the top of the board should start flashing within about a minute to tell you that the module has a valid fix.

### CONNECTING THE LS20031 TO A PC.

Now we need to connect the module to a serial or USB port to configure it and read its output data. The circuit described next can be temporary if it is only used to configure the module for an application other than running on a PC or it can be wired to the module permanently if the module is going to be used with a PC at all times.

The LS20031's TX and RX pins input and output data at 3.3 volts (a data 0 = 0 volts and a data 1 = 3.3 volts). The serial port on a computer is designed to use +/- 12 volts in a differential configuration. To accommodate this difference we will need to employ a line level converter chip. I like the MAX232CPE (Digi-Key Part# MAX232CPE+-ND). You'll also need 4 capacitors (1 MFD at various voltages). If you choose carefully and are not worried too much about physical size then they can all be the same. Examples would be the Vishay/Sprague 199D105X9025A1V1E3 (Digi-Key Part# 718-1209-ND) or the Kemet T350A105K025AT (Digi-Key Part# 399-3528-ND). If you don't have an old serial cable that you can hack apart then you will also need a female DB9 connector: 171-009-203L001 (Digi-Key Part# 209FE-ND) and a cover for it if you like: 956-009-010R031 (Digi-Key Part# 956-09SPGE-ND).

Here is the circuit:



### Bill Of Materials

Qty Reference Part Suggested

- 1 U1 MAX232CPE MAX232CPE+-ND
- 4 C3,C4,C5,C6 1 MFD 25VDC 718-1209-ND
- 1 P1 FEMALE DB9 171-009-203L001
- 1 M1 DB9 COVER 956-09SPGE-ND

Apologies for the squished together BOM. The forum doesn't allow multiple spaces or tabs.

If you don't have a serial port on your PC then you'll need a serial to USB converter cable. "Cables To Go" makes one (part# 26886) that I find reliable for \$30. Install the driver for it and plug it into a spare USB port.

You'll also need to download and install some awesome free software called RealTerm at:

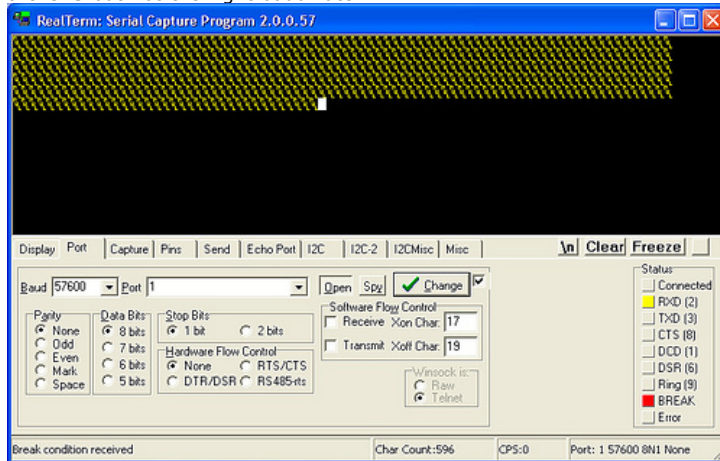
[Terminal Software](#)

You will be using RealTerm to communicate with the LS20031 through the serial port.

Once you've installed RealTerm, plug the DB9 from your line level converter circuit into the serial port on your PC (or the serial to USB converter) and power everything up.

Open up RealTerm.

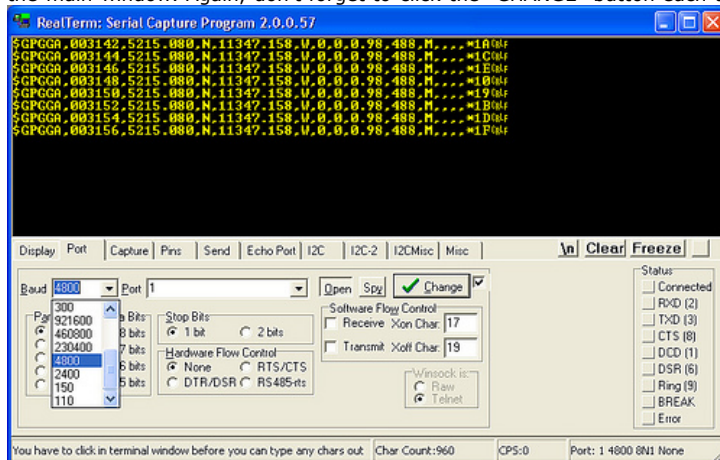
If you see a bunch of unintelligible yellow text scrolling into the main window then it means RealTerm is already set to the correct port and is reading the GPS but not the right baud rate.



If you see nothing in the main window then you need to try the other port numbers until you do. If you have proper NMEA sentences appearing as described in the LS20031's datasheet then go buy a lottery ticket because you were fortunate enough to have all the stars align on the first go.

In order to find the port number that the GPS/Serial to USB adapter is running on got to the "PORT" tab on RealTerm and keep choosing the next available port number until you see text of some kind scrolling in the main window. Don't forget to click the "CHANGE" button each time you choose a different port number.

Once you have found a port that shows text coming in from the GPS then start choosing different baud rates until NMEA sentences begin to appear in the main window. Again, don't forget to click the "CHANGE" button each time you select a new baud rate.



**TIP:** The LS20031 datasheet has a section explaining how to interpret these NMEA sentences.

Once you have proper NMEA sentences showing up in the RealTerm window, if you have map software that you would like to run with this GPS module and the baud rate coming from the module is correct for that software (ie. 4800 bps for S&T) then you can shut off RealTerm and fire up the map software (they can't run together because RealTerm will not release the port for the map software to use).

### CONFIGURING THE LS20031

If you need to change the baud rate that the LS20031 is communicating with, or change the position update rate or the type of NMEA sentence being output, etc. you can do this with RealTerm too.

RealTerm does not save your settings so when you open it again you'll once again have to go to the "PORT" tab and select the port and baud rate as described above. Don't forget to click the "CHANGE" button.

Next go to the "SEND" tab and in the EOL section check mark the first +CR and the first +LF. RealTerm is now ready to send configuration commands to the LS20031.

The LS20031 like many GPS modules uses PMTK protocol to receive configuration commands.

Some of the commands are published in various locations and some you will have to assemble on your own. For instance the command to change the

modules communication baud rate to 4800bps is \$PMTK251,4800\*14. The command to change it to other baud rates is not listed. The "14" at the end of the command is a checksum number which means it needs to be recalculated every time you change the contents of the command so simply changing the "4800" to another value will not work. For instance the command to change the baud rate to 38400bps is \$PMTK251,38400\*27. It is not difficult but there are a couple of steps involved. I'll explain some shortcuts that will make this a fairly routine exercise.

**TIP:** I have no idea what will happen and am not responsible for you entering commands which ask the module for configurations which are outside its operating specifications. Many of the module's operating limits are not specified by the manufacturer.

There is a brief technical description of the PMTK protocol and some generic commands here:

<http://www.robotshop.ca/content/PDF/...-gps-08975.pdf>

Some other commands I've gleaned from other sources and I'll list what I have found which will get you by in most cases.

So let's look at how a command is built. We'll use the command to change the baud rate to 4800bps again as an example:

\$PMTK251,4800\*14

**\$** This indicates the beginning of a sentence to the GPS. It never changes.

**PMTK** This indicates the type of sentence to the GPS ie. "This is a command". It never changes.

**251** This is the command type. 251 specifically means "we are going to change the baud rate".

**,** This comma is called a delimiter and is used as separator (in this case to separate the command function from the value).

**4800** This is the value applied to the command, in this case 4800 bits per second.

**\*** another delimiter

**14** This is a checksum number. It is used by the GPS to determine whether or not it has received the sentence accurately. It is created by the command sender (PC in this case) by performing "exclusive or" math on the sentence (whatever the heck that means). Actually it performs the math on the piece of the sentence between \$ and \*. You don't need to know how to do this because there are XOR checksum calculators on line that do it for you. The one I use is here:

[NMEA MTK checksum calculator](#)

So in summery:

\$ = Hello.

PMTK = command is coming.

251, = change the baud rate to:

4800 = 4800 bits per second

\*14 = "exclusive or" of the sentence equals 14. If that's not what ya got then ignore the sentence.

So if we want to change the baud rate to 38400 then we start with:

PMTK251,38400

We put this value into the checksum calculator located here:

[NMEA MTK checksum calculator](#)

and the checksum calculator outputs:

\$PMTK251,38400\*27

Ok, lets do an example with another type of command.

Command 314 is called the "packet type" command and it is responsible for setting up which type of NMEA sentences are output by the module (ie, GGA,RMC etc.) and what rate they are output at.

The LS20031 calculates a position fix 5 times a second. You can configure it to output particular sentence types at different intervals from every update to as infrequently as every 5th update (which would equal about once a second).

**TIP:** If you are running at a low baud rate and try to output every type of sentence at a rate of every update there won't be time for the module to transmit that much information between each update. The manufacturer does not specify what will happen in this case and I don't dare to speculate. To avoid disaster enable only the strings you are interested in and if you need the full 5Hz transmit rate then it would be wise to do some basic baud rate/time frame calculations. Each character uses 10 bits and there is a long stop bit at the end of each sentence. The GGA sentence alone can be over 85 characters in length. Be careful its length is somewhat dynamic.

Lets look at the command sentence:  
\$PMTK314,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0\*28

Each of the zeros represents a particular type of NMEA output sentence.  
The first zero to the left is zero number 0. the last zero to the right is zero number 16

They each correspond with the following type of sentence:

- 0 NMEA\_SENT\_GLL, // GPGLL interval - Geographic Position - Latitude longitude
- 1 NMEA\_SENT\_RMC, // GPRMC interval - Recommended Minimum Specific GNSS Sentence
- 2 NMEA\_SENT\_VTG, // GPVTG interval - Course Over Ground and Ground Spd
- 3 NMEA\_SENT\_GGA, // GPGGA interval - GPS Fix Data
- 4 NMEA\_SENT\_GSA, // GPGSA interval - GNSS DOPS and Active Satellites
- 5 NMEA\_SENT\_GSV, // GPGSV interval - GNSS Satellites in View
- 6 NMEA\_SENT\_GRS, // GPGRS interval - GNSS Range Residuals
- 7 NMEA\_SENT\_GST, // GPGST interval - GNSS Pseudorange Errors Statistics
- 13 NMEA\_SENT\_MALM, // PMTKALM interval - GPS almanac information
- 14 NMEA\_SENT\_MEPH, // PMTKEPH interval - GPS ephemeris information
- 15 NMEA\_SENT\_MDGP, // PMTKDGP interval - GPS differential correction info
- 16 NMEA\_SENT\_MDBG, // PMTKDBG interval - MTK debug information

As the 314 command sentence above is written now (with all zeros) none of the NMEA sentences would be output. Placing a value of 1-5 in any one of the zero positions will turn on the corresponding sentence type and determine its rate.

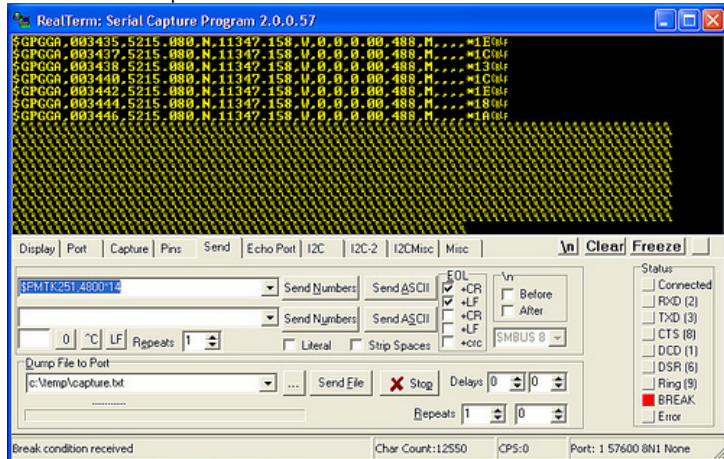
- 0 = Disabled or not supported sentence
- 1 = Output once every one position fix
- 2 = Output once every two position fixes
- 3 = Output once every three position fixes
- 4 = Output once every four position fixes
- 5 = Output once every five position fixes

So if we place a 2 at zero #3 and a 5 at zero #15 like this:  
PMTK314,0,0,2,0,0,0,0,0,0,0,0,0,0,0,0,5,0  
then the GGA string will be output once every 2 position fix updates and the MDGP sentence will be output once every 5 position fix updates (or once a second).

If we put the above sentence into the checksum calculator then we end up with:  
\$PMTK314,0,0,2,0,0,0,0,0,0,0,0,0,0,0,0,5,0\*2F

**TIP:** When inputing a sentence to the checksum calculator make sure you are using all upper case characters (Ascii is case sensitive) and that all spaces are removed from the beginning and end of the sentence.

To send this command to the LS20031 go to the "SEND" tab in RealTerm. Copy the command sentence into the top field beside the "SEND NUMBERS" button and then press the associated "SEND ASCII" button. The text scrolling into the main window should change to reflect the new configuration.



**TIP:** If you change the baud rate configuration (command 251) the text will become temporarily unintelligible (as seen in the image above) until you change RealTerm (in the "PORT" tab) to match the modules new baud rate. Don't forget to click the "CHANGE" button after making a new selection.



Last edited by L6E; Jul 12, 2010 at 02:34 PM. Reason: typos

- [Cell Phone Spy Tools \\$30](#) Secretly monitor text messages, phone calls, GPS Location & more. [www.SignalSpy.com](http://www.SignalSpy.com)
- [InHand Intel® Atom™](#) Ultra-low power embedded SBC based on the Intel® Atom™ Processor [www.inhand.com/product](http://www.inhand.com/product)
- [EZ-Tap RS232 Monitor](#) New compact, low-cost RS232 passive tap module from Stratus Engineering [www.StratusEngir](http://www.StratusEngir)

Jul 12, 2010, 06:27 PM

# 2

t1dunn

**Re: DIY GPS module using the Locosys LS20031**

Wow!, L6E,

This is a bit over my head. But, it is fun to think I might do a project, someday, and rig a R/C helicopter, or something.

What really impresses me is the amount of work you put into writing the post and your willingness to share it. Thanks!

Jul 13, 2010, 03:14 PM

# 3

L6E

**Re: DIY GPS module using the Locosys LS20031**

Thanks.

I hope it's helpful to some.

I was excited to post this in particular for two main reasons. First, almost all GPS modules are very similar in operation to this one so I figured this post would contain helpful technical information for the forum. Second is that it's a very easy project to tackle and has many practical applications. The parts are inexpensive and readily available. Anybody that has hobbled with electronics enough to solder components together can easily manage this.

P.S. For anybody delving in, I'm happy to provide suggestions if you're having trouble with something.

*Last edited by L6E; Jul 13, 2010 at 03:15 PM. Reason: Typo*

Jul 13, 2010, 05:24 PM

# 4

tcassidy

**Re: DIY GPS module using the Locosys LS20031**

I love all the detail too. However, wouldn't it be better for a builder who is using a laptop to go with the USB chip. I assume the line level convertor and serial-usb cable would not be required then. The only problem might be a USB-serial driver for the o/s of the laptop.

Terry

Jul 13, 2010, 06:42 PM

# 5

L6E

**Re: DIY GPS module using the Locosys LS20031**

Hi Terry,

If the application is strictly for a laptop then yes, as I mentioned at the beginning of the thread there is the USB version of this module and if I get involved with it I'll definitely post details on it as well.

I hope I'm not out of line posting information that touches subjects outside of strictly laptop applications. My reason for posting about the raw version is that it explores the raw data that most modules use to communicate with a device whether it be a USB adapter circuit, or serial port, IR device, mobile device, microcontroller or whatever. This is helpful for troubleshooting and for those with odd applications where a USB port is not used. For example I'm working on a post that describes how to hook the GPS module up to a wireless device which provides much greater range than Bluetooth. This is something that is not practical with USB do to propagation delay.

Describing how the raw module functions provides understanding of how a GPS module communicates with its host whether it's a laptop or other device. This could be useful for those exploring if their USB module will accept PTMK commands, a potential hack to open a module up to more functions.

There is a lot of helpful information on how to plug a GPS module into a USB port but I am hoping to expand the knowledge base a bit to those tricky applications that people are still having trouble finding solutions for.

As for the USB version it appears to have a TTL to USB converter built in so getting it running would be a simple matter of finding out if the manufacture has a USB to virtual serial driver made for it or if they expect you to use a generic one (since the PTMK protocol is standard among many GPS modules) or if they expect you to code your own. I'm guessing that with a little trial and error a driver from one of the many other packaged modules out there would work, at least in the GPS to laptop communication direction.

*Last edited by L6E; Jul 13, 2010 at 07:20 PM. Reason: typo*

Jul 13, 2010, 06:47 PM

# 6

Marvin Hlavac

Laptop GPS World  
[www.laptopgpsworld.com](http://www.laptopgpsworld.com)

**Re: DIY GPS module using the Locosys LS20031**

Quote:

*I hope I'm not out of line posting information that touches subjects outside of strictly laptop applications.*

It's still very much on-topic. Thanks for posting it. I'm sure it's a very interesting reading for many of us. 🙌

Jul 13, 2010, 07:57 PM

# 7

tcassidy

**Re: DIY GPS module using the Locosys LS20031**

I would agree the serial module is probably more flexible than the USB one. It sounds like the interface with the transceiver module would be more straight forward too. Do you need the level converter in that case as well?

Terry

Jul 13, 2010, 09:20 PM

# 8

L6E

**Re: DIY GPS module using the Locosys LS20031**

The same line level converter circuit as described above will be required between the device running the map software and the transceiver associated with it. A line level converter is not required at the remote end for the GPS because the associated transceiver is designed to use the same TTL level signals. In fact the transceiver can be configured to automatically handle the NMEA standard too. Its a very strait forward setup. If you need to go into a USB port at the software end then an off the shelf serial to USB converter like the "Cables to Go" one mentioned above will work nicely for both TX & RX communication with the GPS.

Jul 13, 2010, 09:29 PM

# 9

tcassidy

**Re: DIY GPS module using the Locosys LS20031**

Ok, I see now. Basically, you can just - in effect - break the diagram shown at the line level converter input to insert the 2 transceiver devices.

Terry

Jul 13, 2010, 09:39 PM

# 10

L6E

**Re: DIY GPS module using the Locosys LS20031**

Exactly.

Jul 16, 2010, 03:51 PM

# 11

MrUmbra

**Re: DIY GPS module using the Locosys LS20031**

Where did you procure the GPS module?

--- CHAS

Jul 16, 2010, 05:02 PM

# 12

L6E

**Re: DIY GPS module using the Locosys LS20031**

In Canada: Robot Shop:  
[SFE LS20031 32 Channel 5Hz GPS Receiver - RobotShop](#)

In the USA, which I have not tried:  
[SparkFun Electronics - 32 Channel LS20031 GPS 5Hz Receiver](#)

Jul 26, 2010, 02:44 PM

# 13

JuanC

**Re: DIY GPS module using the Locosys LS20031**

Hello!

First of all I want to thank you for your post. Right now I'm working with the mechatronic group of my university. My job is to install this GPS into a helicopter that we are planning to make autonomous and this guide have helped me a lot since it's the first time I'm working with a GPS.

However, I have a question. Can you please give me the list of all of the PMTK protocol commands that you own? I've been having troubles finding it.

I've found this [MTK proprietary manual](#), however if you have more commands please tell me.

Juan

Aug 11, 2010, 11:36 PM

# 14

## L6E

### Re: DIY GPS module using the Locosys LS20031

Hi Juan,  
Sorry for the late reply, been away for a while. Pretty much what I have posted above is all that I have right now. I've seen several other PMTK commands mentioned briefly while searching the web but with no explanation about how they are used. You may try contacting the manufacture of the receiver Locosys or the manufacture of the GPS chipset MediaTek. I'll be doing the same soon but I have to get some other things done first.

Also a Google search for PMTK and Locosys GPS produces other websites where other PMTK commands are being implemented and you may want to contact posters to see and ask where they found them.

I also found this document and although it is for a chipset made by another company, I had a quick glance and the commands I'm familiar with match up so the others listed there are probably relevant too. I have no idea what will happen though if you request a parameter that your GPS is not designed to produce (ie. if you request a baud rate of 4800 and the GPS you are working with is only capable of going down to 9600).

<http://www.rigacci.org/wiki/lib/exe/...sheet-v1.2.pdf>



#### Tags

[diy](#), [gps](#), [locosys](#), [module](#)

#### Similar Topics

- [GPS Tracking of Deer Fawns Directly to Your Laptop! \(Locosys LS20031\)](#)
- [Qualcomm 2nd generation Gobi module improves GPS and mobile Internet for laptops](#)

[Contact us](#) [Resources](#) [Site Map](#)

© Laptop GPS World, 2007-2011