GPS PROJECT

In early July I purchased one of John Mavin's bare-board GPS receivers. He was selling them at his booth at the bay area's Foothill College flea market. With a few hours effort, I was obtaining useable GPS data (time, date, satellite status, and location) from the device. For me, potential applications for the unit are use as an accurate clock for the 'shack', APRS (Amateur Packet Reporting System), and possibly interfacing the unit to existing PC-based travel/map software.

The basic steps to get the unit 'operational' are (1) attach the supplied magnet mount antenna, (2) apply 12VDC power, (3) hook-up the data-out output to a PC using RS-232 converter, and (4) write and 'run' a simple BASIC program to receive and decode the data. The following describes these steps in more detail.

The first good sign is that the GPS unit is neatly packed in bubble-wrap and inside the manufacturer's intended packing box. Most likely this means that the GPS unit had passed some sort of factory test and is not a test reject. The second good sign was the availability of both a hardware and communication specification (check out http://mavin.com, they exist there in 'pdf' format).

The Japanese Aisin Seiki Company manufactured the board. A check of their web site discloses that they sell various consumer products for automobiles in Japan. Automotive GPS systems are among their product line.

<u>Antenna and Power</u>

After plugging in the antenna, the next step was to figure out what type of voltage inputs the unit requires. Since it was designed for automobile operation I understood the 10.5VDC to 16.0VDC specifications, but did not understand the need for 5VDC. Examination of the PCB revealed surface mount filters (FL1 through FL5) for both the input power and the communication I/O. The absence of the surface mount filter (FL2) for the +5VDC power was noticed. Since the unit contains a 7805 regulator I decided to give it a shot with only the 12VDC input. Sure enough, when 12VDC was applied to the appropriate pins (+12VDC at CN1-1 and Ground at CN1-2), the onboard side of the missing FL2 'solder pad' indicated 5VDC.

The next check was to crudely examine the communication output (CN1-5 to ground). Even with a DVM, it was apparent that every second some sort of communication was being broadcast. The next step; build a 5VDC "level" to RS-232 converter so that I can monitor the data using my PC. In the present form the serial output is the wrong polarity and not at the right voltage levels for a PC.

RS-232 Level Converter

The diagram shown in Figure 1 shows the GPS to PC hook-up and includes the signal level converter circuit that I used. The ICL232 (or MAX232, LT1081, TSC232, or DS14C232 – they are all pin compatible) produces the needed RS-232 levels (i.e., the +/- 10V signals) internally using only a 5VDC source (stolen from the aforementioned FL2 ‘solder pad’). The four capacitors are all polarized 2.2uF.

All connections to the GPS engine are via the CN1 connector, the eight pin edge connector. I just soldered my connections directly to the bottom of the printed circuit board.

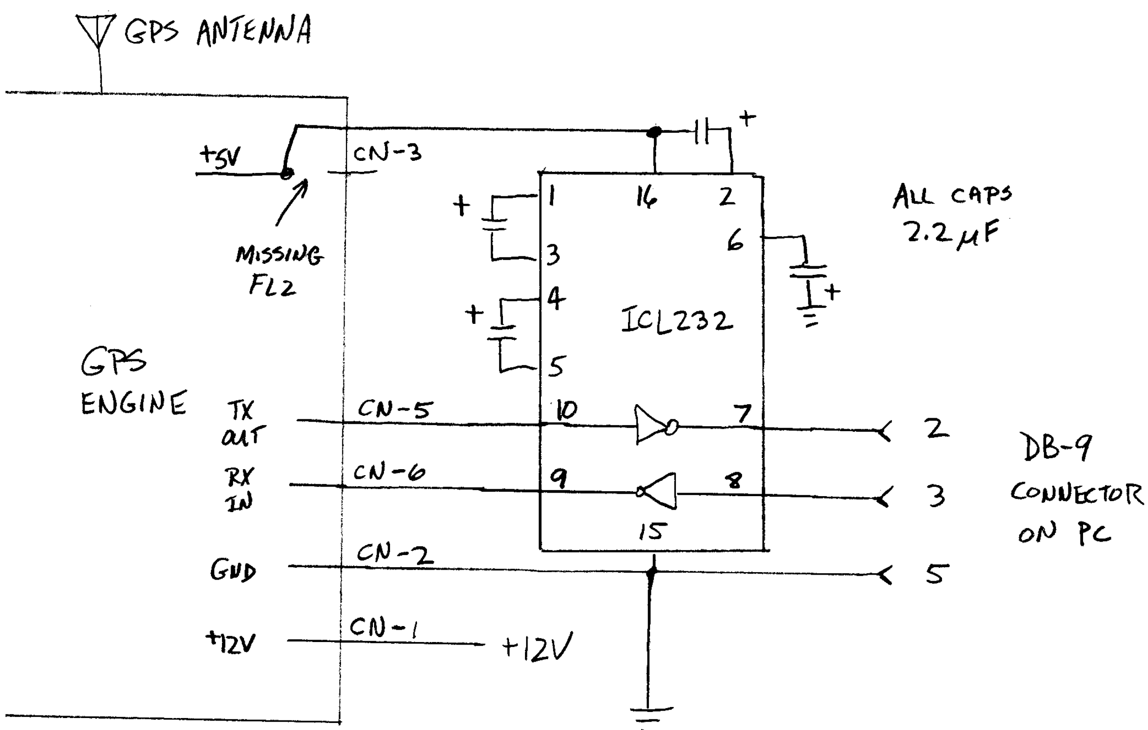

Figure 1

BASIC Program

I recommend the program in Listing #1 for starters. It fills the screen with the hex bytes being received from the GPS unit. The program in Listing #2 decodes the received latitude and longitude.

Program Listing #1 (Basic RS-232 Receive and Display in Hex)

```
    CLS: COL = 16

    OPEN "COM1:9600,N,8,1,CD0,CS0,DS0,OP0,RS,TB100,RB100" FOR RANDOM AS #1

10  NEW = ASC(INPUT$(1, #1))
    GOSUB 770
    IF INKEY$ <> "" THEN 999
    GOTO 10

770 REM ***** Print Hex message *****
    IF NEW < 16 THEN a$ = "0" + HEX$(NEW) + " " ELSE a$ = HEX$(NEW) + " "
    PRINT a$;
    RETURN
    i = i + 1: IF i < COL THEN RETURN
    i = 0
    PRINT
790 RETURN

999 END
```

Program Listing #2 (Decodes Latitude and Longitude)

```
    CLS: COL = 16: DIM A(100)

    OPEN "COM1:9600,N,8,1,CD0,CS0,DS0,OP0,RS,TB100,RB100" FOR RANDOM AS #1

10  A(1) = ASC(INPUT$(1, #1))
    IF A(1) = 16 AND OLD = 3 THEN GOSUB 100
    OLD = A(1)
    IF INKEY$ <> "" THEN 999
    GOTO 10

100 FOR I = 2 TO 45
    A(I) = ASC(INPUT$(1, #1))
    NEXT I

    REM Only Good For North Latitudes (i.e., 0 to +90)
    LAT = (A(8)/256 + A(7) + 256*A(6) + 65536*A(5))/3600
    PRINT "Lat  ="; LAT

    REM Only Good For West Longitudes (i.e., 0 to –180)
    LNG = -4660.337778# + (A(12)/256 + A(11) + 256*A(10) + 65536*A(9))/3600
    PRINT "Long = "; LNG

    RETURN

999 END
```

When the unit was powered up and the output monitored, the following hex string was received (once a second). This message appears to somewhat match the message specified on sheet 34 of the communications document.

10 BF 28 00 01 FA 40 00 07 78 80 00 00 64 00 00 00 00 00 00 00 00 00 00 00
99 07 19 04 46 04 00 00 1B A0 42 14 A0 00 00 00 00 00 3B 10 03

I came to the following conclusions.

First, even though the unit has ‘on board’ battery-backed-up memory, it appears that the on-board clock (in GMT) has been synched by a satellite. The 99 07 19 04 46 04 is interpreted as July 19, 1999, 4 hours, 46 minutes, and 4 seconds. This matched up (to the second) with WWV signal on 10MHz. If nothing else, I now have a board that is capable of auto-updating my computer clock time.

Second, from the message (and using the communication specification, sheet 35), the latitude and longitude is determined to be 36.0000N (from the 01 FA 40 00) and 136.00000E (from the 07 78 80 00). This is somewhere in Japan!

I do not own a commercial GPS, but have been told that when you turn it on in another part of the world, you must either give it a good guess or allow the GPS several minutes to figure out where it is. Sure enough, after about ten minutes, 37.25487N and 121.9013W was decoded and I began to see minor updates in the least significant digit.

Next

The message I am receiving is the general status message. The communication documents specified about 15 ‘query’ type messages and their responses. My next step will be to attempt to command the GPS to send me some of the other messages. These messages include the capability to send a ‘seed’ latitude and longitude (although my GPS unit now powers up with the last power-down location as the ‘seed’), and messages to get other types of data about the satellites (e.g. signal strength, operability, etc.).

Good Luck,

Ken