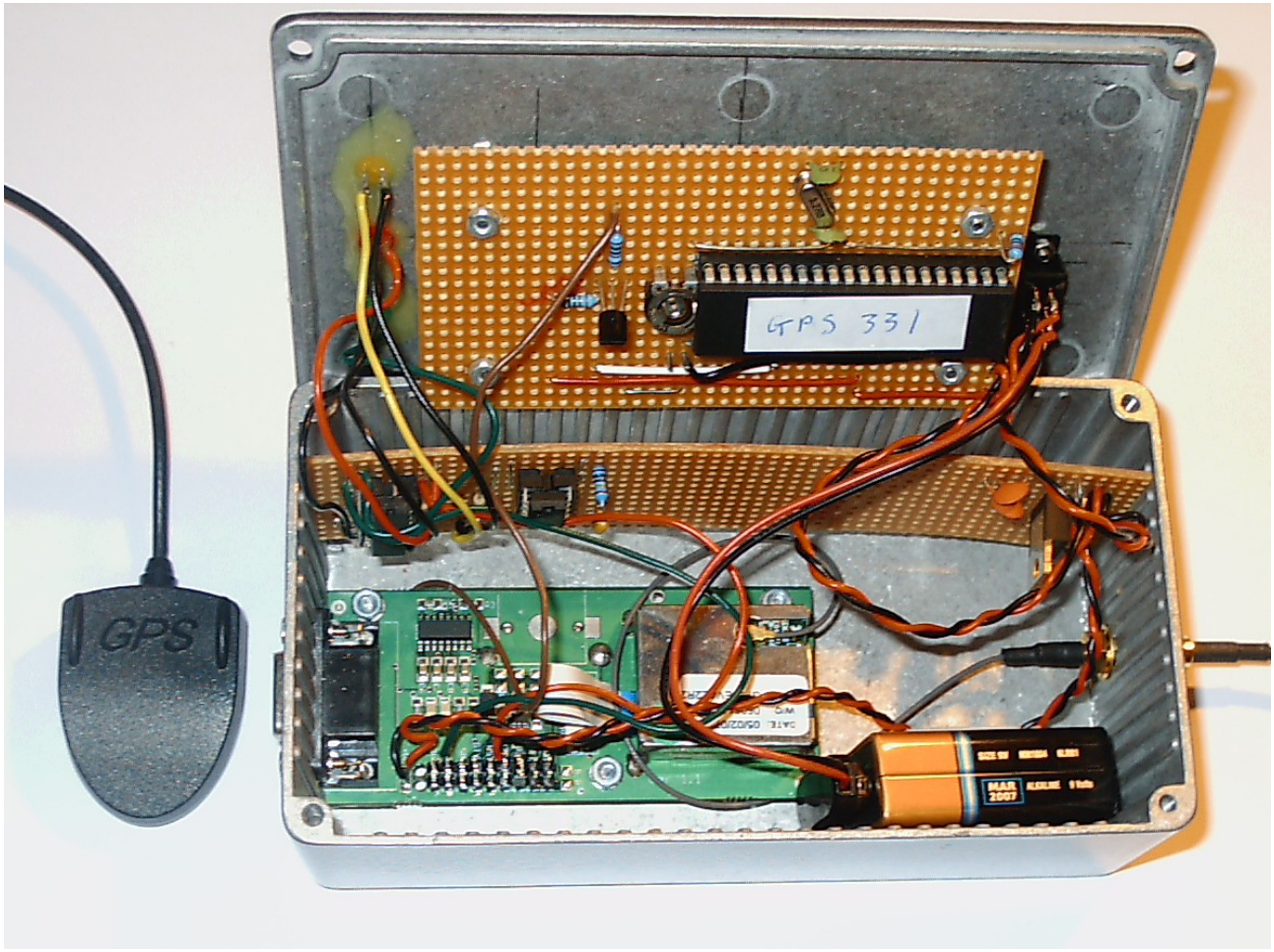
GPS Program Development and Experience

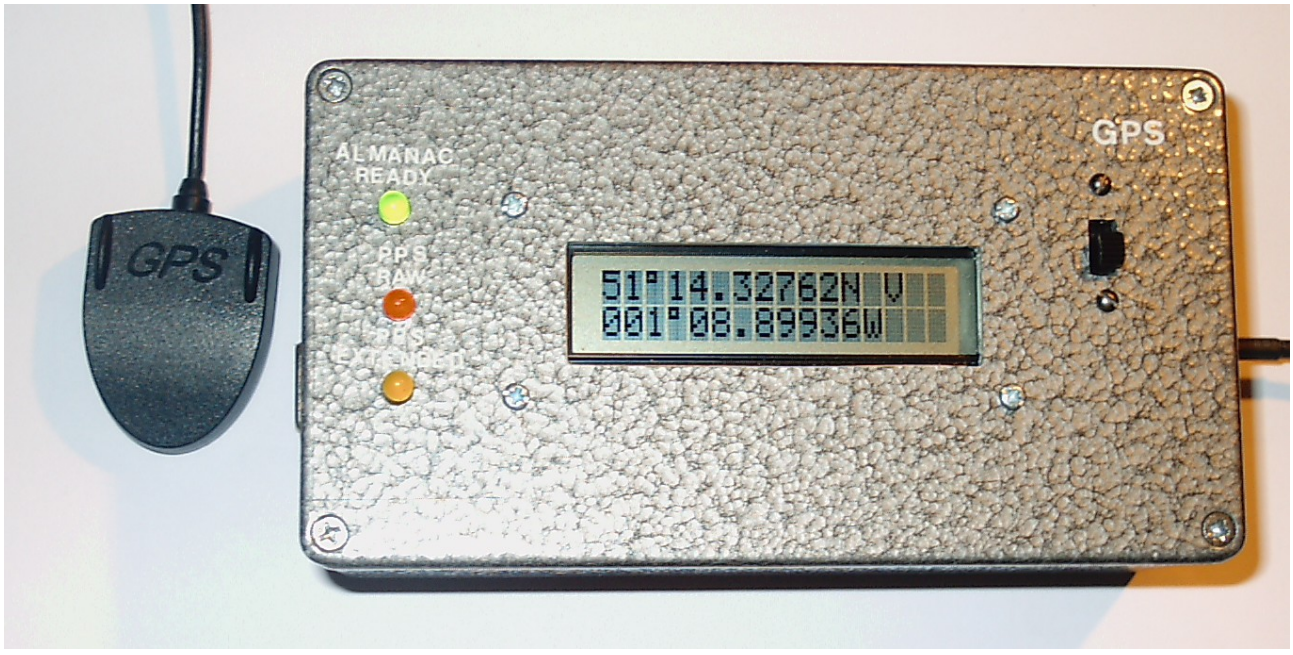
Originally Published in QL Today Vol 13, Issue 1, Sept-November 2008

I have no idea many people have tried Hugh Rooms GPS program or for that matter how many people may be interested in this line of development. But to my mind this is an ideal area to play with, and I think that is what the QL community is all about, experimenting and tinkering. Hugh's original series about GPS was published in Vol 4 Issues 2,3 and 4. At Hugh's own admission, what was presented was not a finished program. Clearly Hugh had developed his program for his own needs, which is fine. What I want to show in this article is what can be done, and my experience of using two differing GPS receivers. I also think this is at a timely point, with the latest series of articles about mapping, Mapping Ancient and Modern by Geoff Wicks, in Vol 12 Issue 2. Now at the time of writing I have not seen the mapping software promised in Geoff's article, so I do not know how my development is going to fit in.

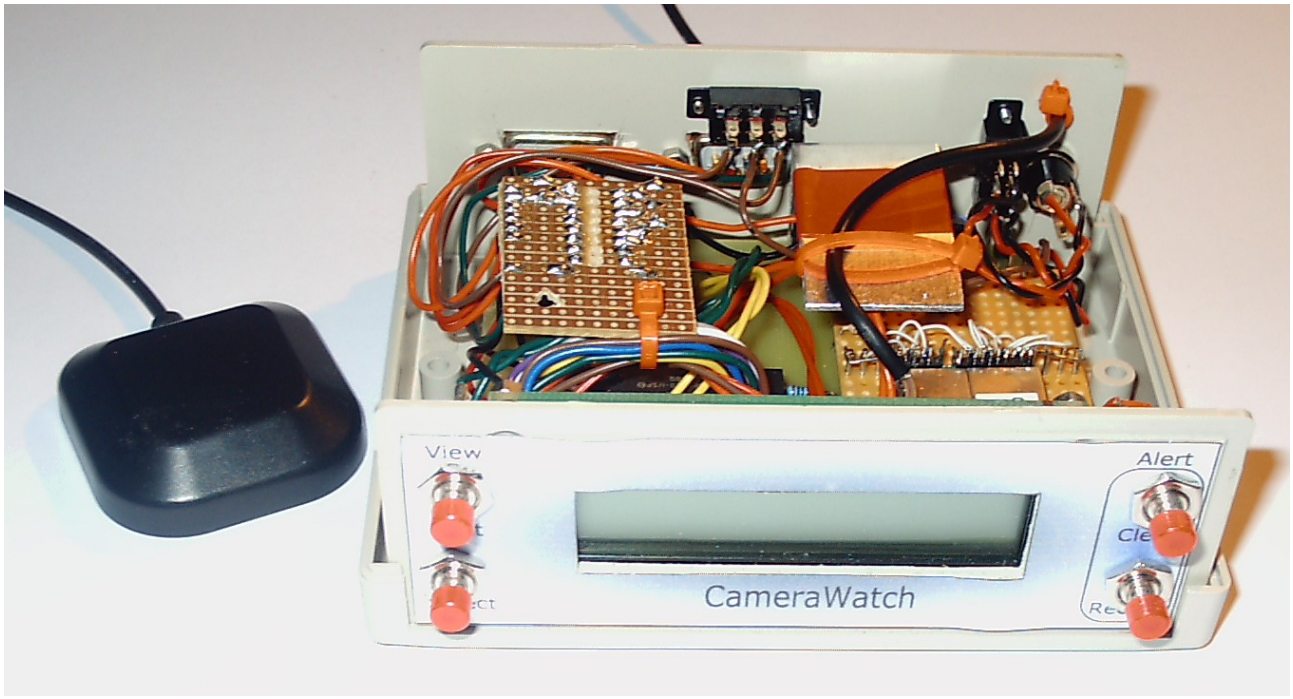
The History

A bit of history. I had already built the Camera Watch device that was published in Everyday Practical Electronics, November 2005 issue, some time ago. I made a small modification to this unit so I could get raw GPS data from the inbuilt receiver to feed my QL system. I used a WD-G-ZX4120 receiver for this project, it is available from Crownhill Associates and at the time of writing was £35.19 including VAT. One of the aims of this article is to show what differences there can be between receivers. There is datasheet for this module on the Crownhill web site which contains all the protocol information. After reading Hugh's article I thought it may be interesting to build a simple unit similar to the one Hugh had built. Taking note of the problems Hugh had with the wiring of the RF Solutions GPSM001 module I purchased the RF Solutions Evaluation Board. Yes it was more expensive than the receiver module on it own, but is comes with an RS232 port and PC software, which is interesting in it's self. So now I had two receivers. Like Hugh I also built and modified the the January 2004 EPE GPS to PIC and PC project so I could use the Evaluation Board on it's own and display longitude, latitude, altitude and time. Like Hugh I had to modify the PIC code due to the variation in the latitude and longitude data as will be explained below.





The first picture above shows the RF Solutions development board fitted in a metal case with the PIC display and the LED's display as is Hugh's original article. I also liked the flashing lights, sad I know. The second picture show the complete unit working.



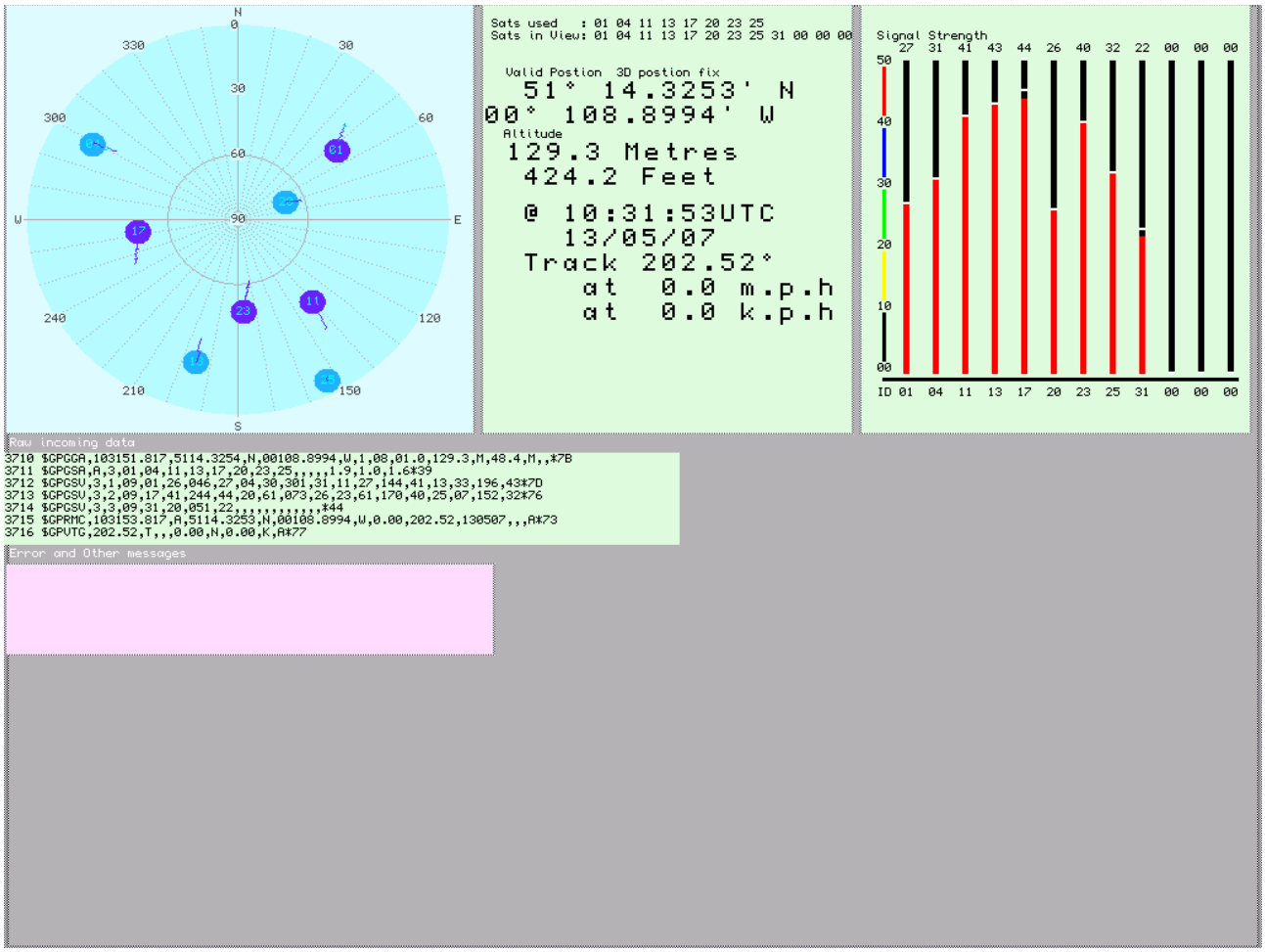
The picture above is my Camera Watch unit. You can see the ZX4120 GPS module, it's the silver box with the bar code on it towards the bottom right hand side. You may be able to see the very fine wiring, not for the faint hearted. I mounted the module on a piece of strip board with double sided tape and short jumpers from the module to the strip board, then wired from the strip board to the main Camera Watch PCB. Also since the ZX4120 required 3.3 volts to power it, there a power regulator on the strip board as well. This is fed from the 5 volts on the main PCB. The main PCB has it's own 5 volt regulator. So the entire unit can be run from 9-15 volts. The original project was designed for use in a car. On the top left of the unit there is another piece of strip board which is up side down it has a 3.3V to RS232 level converter on it. It takes the data from the GPS receiver and transmits it to the serial port of the PC/QL. One small point of warning with these modules, don't plug or unplug the antenna while they are powered up. The antenna, which are the small black boxes to the left in the pictures above, are fed power from the receiver modules. There is an amplifier in the antenna which requires a supply of power. If you accidentally short the antenna connector you could damage the module. So play safe and switch off before connecting or disconnecting the antenna.

The Software

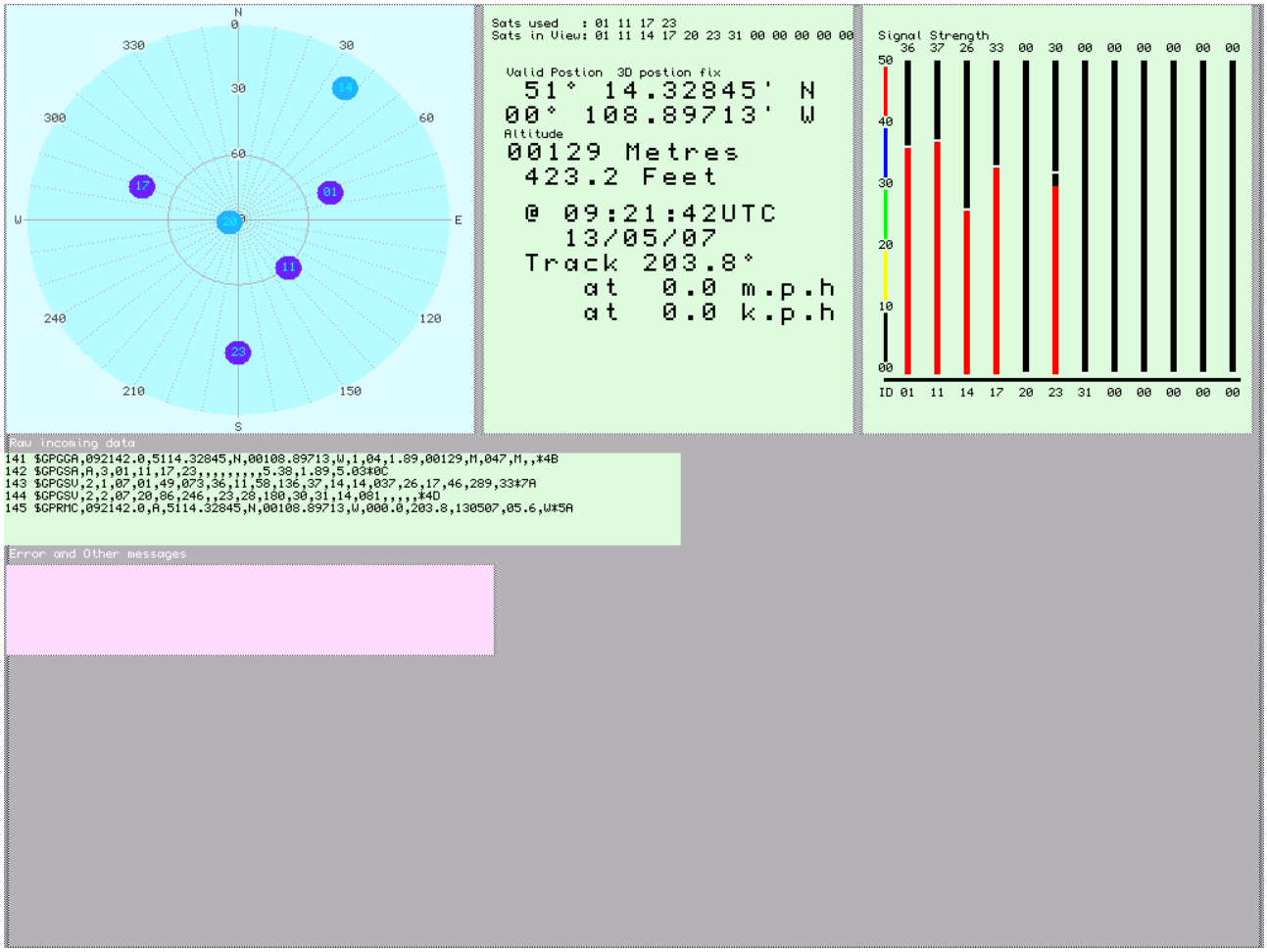
I typed in Hugh's program and got in running. But found some issues when I tried it with both of my GPS receivers. The first being the data stream from the two receivers were not the same. On further investigation there were significant differences. You would have thought the protocol from two GPS receivers would be the same. Well they are not. Also I am sure there are other variations out there. So what were the differences that I found. First point to make was that the order of data within sentences are the same. However fields within sentences can differ. The first one was the longitude and latitude data in the $GPGGA sentence in the case of the ZX4120 module the data has 4 decimal points and the RF Solutions module 5 decimal points. Not a major problem but a difference which will change the way and space you need to display the data. Which can make the resultant display untidy. The second difference was the the ZX4120 module issues up to three $GPGSV sentences, were the RF Solutions module only issues two. The third difference is an additional sentence called $GPVTG. The $GPGSV sentence provides the following data. The number of messages, this can be upto three, but changes over time. Depending on the number of

satellites in view. So sometimes you are receiving three $GPGSV sentences and at other times only two sentences. Never seen it go to one, that is not to say it could not. The next element is the Message number, so you now know how many $GPGSV sentences to expect and which sentence is which. The next piece of data is the number of 'Satellites in View', this is what determines the number of sentences since it is a function of the number of satellites in view. That function is carried out within the GPS module itself. The remaining data in each $GPGSV sentence contains details of each satellite in view upto a maximum of 12, with 4 in each sentence hence the three sentences. So when only 8 or less satellites are seen then the receiver only returns two $GPGSV sentences. The rest of the sentence contains the satellite ID, Elevation, Azimuth and Signal to Noise Ratio of each satellite in turn. So the first $GPGSV sentence returns data for the first four satellites the receiver can see, the second the next four, and then the third the remaining up to 12. An interesting point is the datasheet states in the specification for the module that it can track up to 16 satellites, however the protocol only shows support for 12 satellite. There would have to be 4 $GPGSV sentences to support this and as we have seen there are only a maximum of 3. Also in practice, I have only seen 3, so I have only provided for 3. But it would not to difficult to make it 4 should the need arise. The additional sentence from the ZX4120 module $GPVTG contains course and speed information. In the case of course it returns true and magnetic headings. Speed data is in both knots and K.P.H. Hugh's original program was not designed to deal with the variations in the $GPGSV data in a dynamic way, or the extra $GPVTG sentence. So I set to work. The basic's of Hugh's program was spot on and is very good at trapping errors, in fact too good. In Hugh's original program he quit out of the main loop when ever there was a blank field, or a null field if you like. On the face of it seems a good idea. However the output from the receiver can have valid data after the null field. However you can skip past the blank, null field and there is still good data to be used further along the sentence. Also remember that the sentence has had a check sum preformed on it so it is reasonable to assume that all the data in the sentence is correct. I also wanted to make it more of a finished program so for example I developed a front end. Admission here, I have used a part of Dilwyn Jones DateSet program using a procedure called BANNER (lines 1450 to 2400) which I have modified a little. I just like the Banner effect and the drop shadow. I re-ordered things into what I think is more easy to read form, a more process orientated order as well, so I did not keep jumping around the place while I was adding my own features and broke some areas down into more procedures. I also wanted to display more data than the original. Things like altitude, 2D fix and 3D fix this can happen with out you knowing when you want to resolve altitude. I also added a new window which contains the signal strength information for each satellite, both as a numeric value and as bargraphs with a peak hold feature. The peak hold stays for 3 passes of data (this can be changed to any value you like in variable holdtime% in line 9930), unless a high value comes along in which the peak moves up and resets the hold time. If the level stays below the peak valve then it resets to the new value after the hold time has expired. So a bit like some electronic audio bargraph level meters. In fact this is a routine I have been developing for another application, so I just reused it. This is why the procedure has 13 parameters, I have 'Remarked' the use of each parameter within the program, even though strictly speaking they are not needed in this application. The BARGRAPH procedure is between lines 12480 and 12740. One major change I made was to decode all the data sentences completely, so I did not have to go back at a later date, if I want to make any further changes to other parts of the program (lines 6970 to 9400). It also makes it easier for other users to adapt the program to their own requirements. I also made changes to the raw data display code so it could handle the changing format of data and keep it tidy. Also keeping the line count correctly updated as well, since the line count changes when 3 $GPPSV sentences are received. As well as the $GPVTG sentence being present also effects the line count which is handled automatically as well. I did have to read the $GPGSV sentence on the fly so to speak, so as to determine how many $GPGSV sentences were present on each pass. Remember this changes dynamically (lines 5490-5500). I also, before decoding on-line so to speak, tested the incoming stream to see if the $GPVTG sentence is present or not, with the procedure 'test_receiver_sentence' (lines 6250 to 6350). By automating the process the user does not have know which type of receiver

is being used.



Results from Camera Watch



Results from RF Solutions Development Card

Other changes I made were to the display layout, however it is easy to move the windows around. In fact the error window at presently does not have any messages sent to it, it was in Hugh's original, I left it in because it may be useful in the future. The window containing the longitude, latitude, satellites in view,satellites used, time, speed and course. I added altitude in feet and metres, speed is now in MPH as well as KPH. KPH is the raw data as received. 2D and 3D fix information is also shown and the date has been added under the time field. If you wanted, you can display both the orbit display and track display at the same time, but I have not done this yet. You will find there are some test lines that I have left in, they were used during debugging and testing the decoding process, you can leave them out if you wish. I have left them in for any future development purposes. I am not saying my program is perfect, and I am sure their are people out there that could do a better job of it than me. I was after the features I wanted, not programming elegance. It all adds to the GPS and QL story. So what next, applying the tracking data to a map, may be moving map display. Also track profile by altitude are possibilities. I am sure there are lots of other things that can be done.

References

Dilwyn Jones, Setdate www.dilwyn.uk6.net/misc/index.html
Everyday Practical Electronics GPS PIC to PC and Camera Watch January 2004, November 2005 www.epemag.com
QLToday Hugh Rooms and Geoff Wicks original GPS and Mapping articles. Vol 11, issues 2,3,4 and Vol 12, issue 2

RF Solutions GPS Evaluation Card www.rfsolutions.co.uk
Crownhill Associates Module available and ZX4120 datasheet downloadable in PDF form from www.crownhill.co.uk

The Program

```
10 COLOUR_PAL
100 start_screen
110 init
120 setup_display_windows
130 GPS_data_init
140 data_for_sim
150 main_loop
160 STOP
1000 REMark
********************************************************************************
******************************
1010 DEFine PROCedure start_screen
1020 WINDOW#0;SCR_XLIM,SCR_YLIM,0,0:BORDER#0;2,0,1:PAPER#0;0:INK#0;7:CLS#0
1030 WINDOW#1;SCR_XLIM,SCR_YLIM,0,0:BORDER#1;2,0,1:PAPER#1;12:INK#1;1:CLS#1
1040 WINDOW#2;SCR_XLIM,SCR_YLIM,0,0:BORDER#2;2,0,1:PAPER#2;12:INK#2;1:CLS#2
1050 BANNER 1,(SCR_XLIM/2)-100,50,4,1,4,"QLToday GPS"
1060 INBANNER 1,20,100,1,4,1,4,"Input from GPS (R)eceiver or (F)ile :"
1070 IF in$<>"R" AND in$<>"r" AND in$<>"F" AND in$<>"f" THEN GO TO 1060
1080 IF in$=="f" THEN sim=1:AT#1;7,57:BANNER 1,20,100,4,1,4,"Input from GPS (R)eceiver or
(F)ile : Input from file"
1090 IF in$=="r" THEN sim=0:AT#1;7,57:BANNER 1,20,100,4,1,4,"Input from GPS (R)eceiver or
(F)ile : Input from receiver"
1100 INBANNER 1,20,150,1,4,1,4,"Show (O)rbits or (T)rack :"
1110 IF in$<>"O" AND in$<>"o" AND in$<>"T" AND in$<>"t" THEN GO TO 1100
1120 IF in$=="t" THEN ShowTrack%=1:BANNER 1,20,150,4,1,4,"Show (O)rbits or (T)rack :
Show Track"
1130 IF in$=="o" THEN ShowTrack%=0:BANNER 1,20,150,4,1,4,"Show (O)rbits or (T)rack :
Orbits"
1140 IF ShowTrack%=0 THEN INBANNER 1,20,200,1,4,1,4,"Show Orbits as (B)lobs or
(L)ines :":ELSE GO TO 1180
1150 IF in$<>"B" AND in$<>"b" AND in$<>"L" AND in$<>"l" THEN GO TO 1140
1160 IF in$=="B" THEN Blobs%=1:BANNER 1,20,200,4,1,4,"Show Orbits as (B)lobs or (L)ines :
Show Orbits as Blobs"
1170 IF in$=="L" THEN Blobs%=0:BANNER 1,20,200,4,1,4,"Show Orbits as (B)lobs or (L)ines :
Show Orbits as Lines"
1180 IF sim=0:AT#1;13,20:INBANNER 1,20,250,1,4,1,4,"Save raw data from recevier
(Y/N) :":ELSE GO TO 1260
1190 IF in$<>"Y" AND in$<>"y" AND in$<>"N" AND in$<>"n" THEN GO TO 1140
1200 IF in$=="y" THEN dtof=1:BANNER 1,20,250,4,1,4,"Save raw data from recevier (Y/N) :
Saves raw receiver data"
1210 IF in$=="n" THEN dtof=0:BANNER 1,20,250,3,1,4,"Save raw data from recevier (Y/N) :
Does NOT save receiver data"
1220 IF dtof=1:AT#1;15,20:INBANNER 1,20,300,28,3,1,4,"Enter raw data file name (e.g.
win5_gps_sats_dat) :":ELSE GO TO 1260
1230 raw1$=in$
1240 BLOCK#1,970,25,20,300,12
```

```
1250 BANNER 1,20,300,4,1,4,"Data from receiver will be sent to file :":BANNER
1,680,300,4,7,4,raw1$
1260 IF sim=0:AT#1;17,20:INBANNER 1,20,350,1,4,1,4,"Serial Port GPS Receiver connected to
(1-6) :":ELSE GO TO 1310
1270 SerPort%=in$
1280 IF SerPort%<1 AND SerPort%>6 THEN GO TO 1260
1290 BLOCK#1,800,25,20,350,12
1300 ser$=SerPort%:BANNER 1,20,350,4,1,4,"Receiver connected to serial port :": BANNER
1,593,350,4,1,4,ser$
1310 IF sim=1:INBANNER 1,20,400,28,3,1,4,"Enter raw data file name (e.g.
win5_gps_sats_dat) :":ELSE GO TO 1370
1320 raw2$=in$
1330 BLOCK#1,970,25,20,400,12
1340 BANNER 1,20,400,4,1,4,"Data will come from file :":BANNER 1,450,400,4,7,4,raw2$
1350 REMark Delay in seconds between readings
1360 REMark to avoid enormous sats_data file
1370 INBANNER 1,20,450,1,4,1,4,"Delay between reading receiver data in seconds "
1380 delay=in$
1390 BANNER 1,773,450,4,1,4,delay
1400 INBANNER 1,20,500,1,4,1,4,"Confirm these settings are correct (Y/N)"
1410 IF in$<>"Y" AND in$<>"y" AND in$<>"N" AND in$<>"n" THEN GO TO 1400
1420 IF in$=="n" THEN GO TO 1020
1430 END DEFine start_screen
1440 REMark
**********************************************************************************
**********************************
1450 DEFine PROCedure BANNER(Ch%,Xx%,Yy%,Sz%,Ink%,Pap%,f$)
1460  LOCal l%,X%,Y%,W%,H%
1470  l%=LEN(f$)
1480 SELect ON Sz%
1490  =1:W%=7*(l%+1):H%=19
1500  =2:W%=8*(l%+1):H%=19
1510  =3:W%=12*(l%+1):H%=19
1520  =4:W%=16*(l%+1):H%=19
1530  =5:W%=7*(l%+1):H%=30
1540  =6:W%=8*(l%+1):H%=30
1550  =7:W%=12*(l%+1):H%=30
1560  =8:W%=16*(l%+1):H%=30
1570 END SELect
1580  IF Xx%<0 THEN X%=(512-W%)/2:ELSE X%=Xx%:END IF
1590  IF Yy%<0 THEN Y%=(256-H%)/2:ELSE Y%=Yy%:END IF
1600  BLOCK#Ch%;W%,H%,X%+6,Y%+4,0
1610  BLOCK#Ch%;W%,H%,X%,Y%,0
1620  BANNER_TXT Ch%,Xx%,Yy%,Sz%,Ink%,Pap%,f$
1630 END DEFine BANNER
1640 REMark
**********************************************************************************
**********************************
1650 DEFine PROCedure BANNER_TXT(Ch%,Xx%,Yy%,Sz%,Ink%,Pap%,f$)
1660  LOCal l%,X%,Y%,W%,H%
1670  l%=LEN(f$)
1680 SELect ON Sz%
```

```
1690  =1:W%=7*(l%+1):H%=19:CSIZE#Ch%;0,0
1700  =2:W%=8*(l%+1):H%=19:CSIZE#Ch%;1,0
1710  =3:W%=12*(l%+1):H%=19:CSIZE#Ch%;2,0
1720  =4:W%=16*(l%+1):H%=19:CSIZE#Ch%;3,0
1730  =5:W%=7*(l%+1):H%=30:CSIZE#Ch%;0,1
1740  =6:W%=8*(l%+1):H%=30:CSIZE#Ch%;1,1
1750  =7:W%=12*(l%+1):H%=30:CSIZE#Ch%;2,1
1760  =8:W%=16*(l%+1):H%=30:CSIZE#Ch%;3,1
1770 END SELect
1780  IF Xx%<0 THEN X%=(512-W%)/2:ELSE X%=Xx%:END IF
1790  IF Yy%<0 THEN Y%=(256-H%)/2:ELSE Y%=Yy%:END IF
1800  BLOCK#Ch%;W%-4,H%-2,X%+2,Y%+1,Pap%
1810  OVER#Ch%;1:INK#Ch%;0
1820  CURSOR#Ch%;X%+5,Y%+4:PRINT#Ch%;f$;
1830  CURSOR#Ch%;X%+7,Y%+4:PRINT#Ch%;f$;
1840  CURSOR#Ch%;X%+5,Y%+6:PRINT#Ch%;f$;
1850  CURSOR#Ch%;X%+7,Y%+6:PRINT#Ch%;f$;
1860  INK#Ch%;Ink%
1870  CURSOR#Ch%;X%+6,Y%+5:PRINT#Ch%;f$;
1880 END DEFine BANNER_TXT
1890 REMark
**************************************************************************************
****************************************
1900 DEFine PROCedure INBANNER(Ch%,Xx%,Yy%,Ex%,Sz%,Ink%,Pap%,f$)
1910  LOCal l%,X%,Y%,W%,H%
1920  l%=LEN(f$)
1930 SELect ON Sz%
1940  =1:W%=7*(l%+1+Ex%):H%=19
1950  =2:W%=8*(l%+1+Ex%):H%=19
1960  =3:W%=12*(l%+1+Ex%):H%=19
1970  =4:W%=16*(l%+1+Ex%):H%=19
1980  =5:W%=7*(l%+1+Ex%):H%=30
1990  =6:W%=8*(l%+1+Ex%):H%=30
2000  =7:W%=12*(l%+1+Ex%):H%=30
2010  =8:W%=16*(l%+1+Ex%):H%=30
2020 END SELect
2030  IF Xx%<0 THEN X%=(512-W%)/2:ELSE X%=Xx%:END IF
2040  IF Yy%<0 THEN Y%=(256-H%)/2:ELSE Y%=Yy%:END IF
2050  BLOCK#Ch%;W%,H%,X%+6,Y%+4,0
2060  BLOCK#Ch%;W%,H%,X%,Y%,0
2070  INBANNER_TXT Ch%,Xx%,Yy%,Ex%,Sz%,Ink%,Pap%,f$
2080 END DEFine INBANNER
2090 REMark
**************************************************************************************
****************************************
2100 DEFine PROCedure INBANNER_TXT(Ch%,Xx%,Yy%,Ex%,Sz%,Ink%,Pap%,f$)
2110  LOCal l%,X%,Y%,W%,H%
2120  l%=LEN(f$)
2130 SELect ON Sz%
2140  =1:W%=7*(l%+1+Ex%):H%=19:CSIZE#Ch%;0,0
2150  =2:W%=8*(l%+1+Ex%):H%=19:CSIZE#Ch%;1,0
2160  =3:W%=12*(l%+1+Ex%):H%=19:CSIZE#Ch%;2,0
```

```
2170  =4:W%=16*(l%+1+Ex%):H%=19:CSIZE#Ch%;3,0
2180  =5:W%=7*(l%+1+Ex%):H%=30:CSIZE#Ch%;0,1
2190  =6:W%=8*(l%+1+Ex%):H%=30:CSIZE#Ch%;1,1
2200  =7:W%=12*(l%+1+Ex%):H%=30:CSIZE#Ch%;2,1
2210  =8:W%=16*(l%+1+Ex%):H%=30:CSIZE#Ch%;3,1
2220 END SELect
2230  IF Xx%<0 THEN X%=(512-W%)/2:ELSE X%=Xx%:END IF
2240  IF Yy%<0 THEN Y%=(256-H%)/2:ELSE Y%=Yy%:END IF
2250  BLOCK#Ch%;W%-4,H%-2,X%+2,Y%+1,Pap%
2260  OVER#Ch%;1:INK#Ch%;0
2270  CURSOR#Ch%;X%+5,Y%+4:PRINT#Ch%;f$;
2280  CURSOR#Ch%;X%+7,Y%+4:PRINT#Ch%;f$;
2290  CURSOR#Ch%;X%+5,Y%+6:PRINT#Ch%;f$;
2300  CURSOR#Ch%;X%+7,Y%+6:PRINT#Ch%;f$;
2310  INK#Ch%;Ink%
2320  CURSOR#Ch%;X%+6,Y%+5:PRINT#Ch%;f$;
2330 SELect ON Sz%
2340 =1:=5:CURSOR#Ch%;X%+6+(l%*6),Y%+4
2350 =2:=6:CURSOR#Ch%;X%+6+(l%*8),Y%+4
2360 =3:=7:CURSOR#Ch%;X%+6+(l%*12),Y%+4
2370 =4:=8:CURSOR#Ch%;X%+6+(l%*16),Y%+4
2380 END SELect
2390 INK#Ch%;1:INPUT#Ch%;in$
2400 END DEFine INBANNER_TXT
2410 REMark
*******************************************************************************
*********************************************
2420 DEFine PROCedure init
2430 maxid=30:REMark highest permitted satellite id no.
2440 :
2450 IF ShowTrack%>0 THEN
2460 Minlon=0:Minlet=0:Maxlon=0:Maxlat=0
2470 REMark PRINT "Minlon:";Minlon;" Minlat:";Minlat;" Maxlon:";Maxlon;" Maxlat:";Maxlat
2480 REMark For Displaying track, must set min and max
2490 REMark lat and lon or call a procedure to do so ...
2500 ChiCityMap: REMark May need user input here for required area to be covered.
2510 REMark PfdMap
2520 :
2530 REMark See Jan Jones page 39,40
2540 IF (1+Minlon+Minlat+Maxlon+Maxlat)=1 THEN
2550 CLS
2560 PRINT \\\"   ** Map limits not set ***":STOP
2570 END IF
2580 END IF
2590 :
2600 REMark For orbit display
2610 REMark Colours used for spot showing first observed
2620 REMark position and use of satellite
2630 seentint%=194:usdtint%=96
2640 :
2650 CLS#0:CLS#1
2660 CSIZE#1;0,0
```

```
2670 :
2680 set_serial_port
2690 END DEFine init
2700 REMark
*********************************************************************************
***********************************************
2710 REMark Follow a series of definitions of charts for tracks
2720 REMark Maxlon necessary for lat, lon grid
2730 DEFine PROCedure ChiCityMap
2740 Minlat=50+49/60
2750 Maxlat=50+51/60
2760 Minlon=-47/60
2770 Maxlon=-43/60
2780 END DEFine ChiCityMap
2790 REMark
*********************************************************************************
*****************************************************
2800 DEFine PROCedure PfdMap
2810 Minlat=50.7667
2820 Maxlat=51.05
2830 Minlon=-1-3/60
2840 Maxlon=-.6
2850 END DEFine PfdMap
2860 REMark
*********************************************************************************
*********************************************************
2870 DEFine PROCedure set_serial_port
2880 REMark Set up the Serial Port if needed
2890 IF sim THEN
2900 REMark If sim=1(true)
2910 REMark Serial Port simulated by data file
2920 cs%=FOP_IN(raw2$)
2930 :
2940 ELSE
2950 REMark Otherwise real time data from receiver
2960 REMark SerPort% set to 1 to 6 as selected by the user in the Start_Screen procedure
2970 BAUD SerPort%,4800
2980 SerPort$=SerPort%
2990 cs%=FOPEN("srx"&SerPort$&"IA"): REMark I=Ignore flow control, A=<CR><LF>is end of
line, <CR><FF>is end of page.
3000 END IF
3010 END DEFine set_serial_port
3020 REMark
*********************************************************************************
*********************************************************
3030 DEFine PROCedure setup_display_windows
3040 REMark Window to display raw data
3050 dd%=FOPEN("con")
3060 CURSOR 0,350
3070 PRINT "Raw incoming data"
3080 WINDOW#dd%,550,75,0,365
3090
```

```
 BORDER#dd,1,9,1
3100 PAPER#dd%,36:INK#dd%,0:CLS#dd%
3110 :
3120 REMark Windowto display error and other messages
3130 REMark in particular, corrupted data
3140 de%=FOPEN("con")
3150 CURSOR 0,440
3160 PRINT "Error and Other messages"
3170 WINDOW#de%,400,75,0,455
3180 BORDER#de%,1,9,1
3190 PAPER#de%,39:INK#de%,0:CLS#de%:REMark paper was 36
3200 :
3210 REMark Window for main display of orbits or track
3220 asprat=.8:REMark aspect ratio: width/height
3230 size=3.5:REMark for early fiddling with windows
3240 dc%=FOPEN("con")
3250 Mctr=COS(RAD((Minlat+Maxlat)/2))
3260 High=100*size
3270 wide=137*size*asprat
3280 WINDOW#dc%,wide,High,0,0
3290 BORDER#dc%,1,9,1
3300 INK#dc%,0
3310 PAPER#dc%,37:CLS#dc%
3320 :
3330 REMark Set up for track display
3340 IF ShowTrack%<>0 THEN
3350 difflon=Maxlon-Minlon
3360 difflat=Maxlat-Minlat
3370 SCALE#dc%,difflat,Minlat*asprat*Mctr,Minlat
3380 :
3390 REMark Lat and Lon grid
3400 LatLonGrid
3410 :
3420 REMark Window to show instantaneous track and speed
3430 dt%=FOPEN("scr")
3440 Topdc%=3
3450 WINDOW#dt%,137/4*size*asprat,100/4*size,280,Topdc%
3460 BORDER#dt%,1,9,1
3470 SCALE#dt%,200,-100*asprat,-100
3480 PAPER#dt%,36:CLS#dt%
3490 ELSE
3500 REMark For orbit display
3510 SCALE#dc%,2.2,-1.1*asprat,-1.1
3520 REMark set up alternative for orbits
3530 SatSky
3540 END IF
3550 :
3560 REMark Window to show speed, bearing, validity etc.
3570 ds%=FOPEN("con")
3580 WINDOW#ds%,305,350,388,0
3590 BORDER#ds%,1,9,1
3600 PAPER#ds%,36:CLS#ds%
```

```
3610 :
3620 REMark Window to show satellite signal data
3630 dn%=FOPEN("scr")
3640 WINDOW#dn%,320,350,697,0
3650 BORDER#dn%;1,9,1
3660 PAPER#dn%,36:CLS#dn%
3670 INK#dn%;0
3680 AT#dn%;2,2:PRINT#dn%;"Signal Strength"
3690 AT#dn%;4,2:PRINT#dn%;"50"
3700 AT#dn%;9,2:PRINT#dn%;"40"
3710 AT#dn%;14,2:PRINT#dn%;"30"
3720 AT#dn%;19,2:PRINT#dn%;"20"
3730 AT#dn%;24,2:PRINT#dn%;"10"
3740 AT#dn%;29,2:PRINT#dn%;"00"
3750 AT#dn%;31,2:PRINT#dn%;"ID"
3760 BLOCK#dn%;3,40,17,50,2
3770 BLOCK#dn%;3,40,17,100,4
3780 BLOCK#dn%;3,40,17,150,3
3790 BLOCK#dn%;3,40,17,200,6
3800 BLOCK#dn%;3,40,17,250,0
3810 BLOCK#dn%;290,3,17,303,0
3820 DIM peak%(12):REMark area to store peak signal strength.
3830 DIM pht%(12):REMark area to store loop count before reseting peak hold to current level
3840 END DEFine setup_display_windows
3850 REMark
********************************************************************************
************************************************************
3860 :
3870 DEFine PROCedure LatLonGrid
3880 REMark For track, prints a grid of lats and longs
3890 REMark .. each at one minute of acr intervals
3900 REMark parameters set in main program
3910 REMark No need for precise match to window as SB
3920 REMark just doesn't draw outside it
3930 LOCal Glat, Glon, Gld,Glm
3940 INK#dc%,13:REMark nice pale gray
3950 :
3960 REMark Start with latitudes
3970 Gl=Minlat
3980 REMark need to convert to decimal degrees
3990 Gl=INT(Gl)+(INT((Gl-INT(Gl))*60))/60
4000 Gminl=Minlon*asprat*Mctr:Gmaxl=Maxlon*asprat*Mctr
4010 REPeat LatLines
4020 IF Minlat<=Gl AND Maxlat>Gl THEN
4030 LINE#dc%,Gminl,Gl TO Gmaxl,Gl
4040 END IF
4050 Gl=Gl+1/60
4060 IF Gl>Maxlat THEN EXIT LatLines
4070 END REPeat
4080 :
4090 REMark next meridians
4100 Gl=Minlon
```

```
4110 Gl=INT(Gl)+(INT((Gl-INT(Gl))*60))/60
4120 Gminl=Minlon:Gmaxl=Maxlon
4130 REPeat LonLines
4140 IF Minlon<=Gl AND Maxlon>Gl THEN
4150 Gp=Gl*asprat*Mctr
4160 LINE#dc%,Gp,Minlat TO Gp,Maxlat
4170 END IF
4180 Gl=Gl+1/60
4190 IF Gl>Maxlon THEN EXIT LonLines
4200 END REPeat
4210 END DEFine LatLonGrid
4220 REMark
********************************************************************************
**************************************************
4230 DEFine PROCedure SatSky
4240 REMark Sky disk
4250 FILL#dc%,1
4260 INK#dc%,29
4270 ELLIPSE#dc%,0,0,1,1*asprat,0
4280 FILL#dc%,0
4290 :
4300 REMark Draw polar plot grid lines
4310 INK#dc%,12
4320 radials:REMark draw the bearings
4330 REMark Now draw the elevations
4340 FOR i=1 TO 3
4350 ELLIPSE#dc%,0,0,1/3,1*asprat,0
4360 END FOR
4370 LINE#dc%,-1.02*asprat,0 TO 1.02*asprat,0
4380 LINE#dc%,0,-1.02 TO 0,1.02
4390 :
4400 REMark Mark point of compass
4410 INK#dc%,9
4420 CURSOR#dc%,1.03*asprat,0,0,-4
4430 PRINT#dc%,"E"
4440 CURSOR#dc%,-1.03*asprat,0,-6,-4
4450 PRINT#dc%,"W"
4460 CURSOR#dc%,0,1.04,-3,-8
4470 PRINT#dc%,"N"
4480 CURSOR#dc%,0,-1.03,-3,2
4490 PRINT#dc%,"S"
4500 :
4510 REMark Mark azimuth scale
4520 FOR i=30 TO 330 STEP 30
4530 IF (i MOD 90)=0 THEN NEXT i
4540 j=i+90
4550 CURSOR#dc%,1.03*COS(j*PI/180)*asprat,1.03*SIN(j*PI/180),-6,-5
4560 PRINT#dc%,360-i
4570 END FOR i
4580 :
4590 REMark mark elevation scale
4600 FOR i=0 TO 3
```

```
4610 CURSOR#dc%,0,i/3,-6,-5
4620 PRINT#dc%,90-30*i
4630 END FOR i
4640 END DEFine SatSky
4650 REMark
*****************************************************************************
*********************************************************
4660 DEFine PROCedure radials
4670 REMark Draws the celestial meridians at 30 deg intervals
4680 REMark as a series od dots to avoid to dark a line
4690 LOCal i,j,interval
4700 interval=2E-2
4710 FOR i=0 TO 350 STEP 10
4720 FOR j=2 TO 100
4730 REMark uses j*interval as a radial distance ..
4740 REMark .. which must be converted to x,y for plot
4750 IF j*interval>1 THEN EXIT j
4760 POINT#dc%,j*interval*COS(i*PI/180)*asprat,j*interval*SIN(i*PI/180)
4770 END FOR j
4780 END FOR i
4790 END DEFine radials
4800 REMark
*****************************************************************************
*********************************************************
4810 DEFine PROCedure GPS_data_init
4820 :
4830 REMark Display now set up, now get ready for GPS data:
4840 :
4850 REMark Array to store the raw data lines from the reciever
4860 DIM rawdata$(6,128):REMark some receivers may issue more lines, so increase this array as
required
4870 REMark Array to store satellite data..
4880 REMark with: IdNo, Bearing, Elevation, Usage
4890 REMark where: usage is 0 for not used, 1 for used
4900 DIM satvis(12,3):REMark Allow for 12 satellites
4910 :
4920 DIM satsusd(12):REMark List of Ids for satellites used:
4930 :
4940 REMark I use copies of the raw data...
4950 DIM satlist$(6,128):REMark Satellite data from $GPSGV input
4960 REMark Allowed for 7 $GPSGV lines (OTT - never more than 2 !), Not quiet true, is
dependant on receiver, some issue 3 $GPSGSV lines.
4970 :
4980 DIM rmcdata$(128):REMark Lat,Long,Time
4990 :
5000 REMark Posns for orbits plotted as lines
5010 REMark to draw a blob for first point, then a line
5020 REMark Store: Old x,y;New x,y:5th item, )=new orbit...
5030 REMark .. so draw blob, 1=orbit started so draw line
5040 DIM posns(maxid,5):REMark need enough for each satellite, maxid set in line 2410, default
maxid=30
5050 REMark set all to zero for a start
```

```
5060 FOR i%=0 TO maxid-1
5070 FOR j%=0 TO 4
5080 posns(i%,j%)=0
5090 END FOR j%
5100 END FOR i%
5110 :
5120 REMark For plotting trach, need a point to start
5130 REMark then continue with lines.
5140 REMark Don't bother not defining this if orbits
5150 FirstPt%=0
5160 :
5170 Lines=0:REMark A count of input lines
5180 GoodLines=0:REMark Another count of input lines
5190 CLS#dd%:REMark I needed this for raw data display
5200 END DEFine GPS_data_init
5210 REMark
********************************************************************************
************************************************
5220 DEFine PROCedure data_for_sim
5230 REMark Open a file to collect the data from the receiver for simulation
5240 IF dtof<>0 THEN
5250 fc%=FOP_NEW(raw1$)
5260 END IF
5270 END DEFine data_for_sim
5280 REMark
********************************************************************************
************************************************
5290 DEFine PROCedure main_loop
5300 REMark Setting up complete, now for ...
5310 REMark ... data reading and display
5320 :
5330 REMark Repeat loop for continous display
5340 REMark delay at end
5350 REMark Each run through loop deals with a single set
5360 REMark of data sentences, 5 to 7 depending on incoming data stream, starting
5370 REMark with a $GPGGA. Sent each second, but read
5380 REMark at a rate determined by missing some with the delay
5390 REMark Test for $GPVTG sentence present or not, WD-G-ZX4120 module has this sentence,
but the RF Solutions development card does not.
5400 test_receiver_sentences
5410 :
5420 REPeat orbits
5430 AT#dd%;6,0:PRINT#dd%;"                                                    "
5440 AT#dd%,0,0:AT#ds%;0,0:REMark To keep the display tidy
5450 REMark I read all the sentences at one go so that I
5460 REMark don't end up with some from a later second's lot.
5470 rawdata$(0)=gpsdata$("$GPGGA"):REMark wait for a first, GPS fix data
5480 rawdata$(1)=gpsdata$("$GPGSA"):REMark read the rest, GNSS DOP and active satellites
5490 rawdata$(2)=gpsdata$("$GPGSV"):REMark Satellites in view
5500 nummes=rawdata$(2,8)
5510 rawdata$(3)=gpsdata$("$GPGSV"):REMark Satellites in view
5520 rawdata$(4)="$GPGSV,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,*"
```

```
5530 IF nummes=3 THEN rawdata$(4)=gpsdata$("$GPGSV"):REMark Satellites in view
5540 rawdata$(5)=gpsdata$("$GPRMC"):REMark Recommended minimum specific GNSS data
5550 IF VTG%=1:rawdata$(6)=gpsdata$("$GPVTG"):REMark Velocity and track over ground
5560 IF VTG%=1 THEN Lines=Lines+6:ELSE Lines=Lines+5:REMark lines, Goodlines bit messy
...
5570 REMark ... intended it to help look at corrupted data
5580 REMark Check lines for not corrupted data
5590 IF VTG%=1 THEN t=6:ELSE t=5
5600 FOR i=0 TO t
5610 IF i=4 AND nummes<>3 THEN NEXT i
5620 REMark Go through data to look for corrupted lines
5630 REMark This became a long winded process so that is
5640 REMark why I read all the sentences in one go.
5650 REMark Ignoring all the set of data. If there is one
5660 REMark corrupted item, is a bit OTT, but safe.
5670 :
5680 REMark Sometimes get lines much longer than spec..
5690 REMark .. 'Long' lines seem to be a normal line
 but..
5700 REMark .. no <CR><LF> and followed by more,..
5710 REMark .. badly formed, data, so ignore them :
5720 ChkFld%='*' INSTR rawdata$(i)
5730 REMark IF LEN(rawdata$(i))>ChkFld%+3 THEN NEXT orbits
5740 :
5750 IF dtof<>0 THEN
5760 PRINT#fc%,rawdata$(i):REMark Save in file if needed
5770 END IF
5780 :
5790 REMark Check 'Checksum' field
5800 Check%=CheckSum(rawdata$(i))
5810 IF Check%<>0 THEN
5820 NEXT orbits
5830 END IF
5840 GoodLines=GoodLines+1
5850 DisLine dd%,GoodLines&" "&rawdata$(i),80
5860 END FOR i
5870 :
5880 REMark Get ready for GPS fix data
5890 REMark from GPGGA data line, not used in Hugh's orginal program, some data duplicated in
GPRMC data line.
5900 gpsfix$=rawdata$(0)
5910 Extract_gpgga_data
5920 :
5930 REMark Get ready for list of sateliites used
5940 REMark from $GPGSA data line
5950 gpgsa$=rawdata$(1):REMark cautiously us copies
5960 Extract_GPGSA_data
5970 :
5980 REMark Get ready for list of satellite data
5990 REMark First $GPGSV lines gives no of $GPSV lines
6000 satlist$(1)=rawdata$(2)
6010 Extract_GPGSV1_data
```

```
6020 REMark Extract number of $GPGSV lines
6030 novrecs%=numsatview1$
6040 :
6050 REMark Extracting the data from the second $GPGSV sentence
6060 satlist$(2)=rawdata$(3)
6070 Extract_GPGSV2_data:
6080 :
6090 REMark Extracting the data from the third $GPGSV which a dummy setence when not
present in the data stream.
6100 satlist$(3)=rawdata$(4)
6110 Extract_GPGSV3_data
6120 :
6130 REMark Copy RMC data
6140 gpsrmc$=rawdata$(5)
6150 Extract_GPSRMC_data
6160 :
6170 REMark Copy VTG data, if present
6180 gpvtg$=rawdata$(6)
6190 REMark if VTG%=1 then Extract_GPVTG_data
6200 :
6210 display_data
6220 END REPeat orbits
6230 END DEFine main_loop
6240 REMark
**************************************************************************
**************************************************************
6250 DEFine PROCedure test_receiver_sentences
6260 VTG%=0:rawdata$(5)="                                    "
6270 FOR ii=0 TO 7
6280 IF EOF(#cs%) THEN
6290 PRINT "End of File"
6300 CLS#1:CLS#0:CLS#2:CLOSE:STOP
6310 END IF
6320 INPUT#cs%;t$
6330 IF t$(1 TO 6)="$GPVTG" THEN VTG%=1
6340 END FOR ii
6350 END DEFine test_receiver_sentences
6360 REMark
**************************************************************************
**************************************************************
6370 DEFine FuNction gpsdata$(id$)
6380 REMark Waits for and reads a sentence of data starting
6390 REMark with the string id$ (not sat id this time)
6400 REMark Give up if there's no data to serial port
6410 FOR i=1 TO 50
6420 IF EOF(#cs%) THEN
6430 AT#ds%;30,3:PRINT#ds%;"Lines: ";Lines!;
6440 PRINT#ds%;" Good Lines: ";GoodLines
6450 INPUT#ds%;"   Press <enter> to finish: ";t$
6460 CLS#1:CLS#0:CLS#2:CLOSE:STOP
6470 END IF
6480 INPUT#cs%,t$
```

```
6490 REMark More check for dodgy data
6500 IF LEN(t$)=0 THEN NEXT i
6510 IF t$(1)<>"$" THEN NEXT i
6520 IF t$(1 TO 6)=id$ THEN RETurn t$:
6530 END FOR
6540 IF i>10 THEN PRINT#ds%;" No GPS data":STOP
6550 END DEFine gpsdata$
6560 REMark
*****************************************************************************
*************************************************************
6570 DEFine FuNction CheckSum(a$)
6580 LOCal i,aa$,ChkSum,ChkCode$
6590 IF LEN(a$)=0 THEN RETurn -2
6600 FOR i=1 TO 256:REMark Allow for sentence..
6610 REMark .. not terminated properly, i.e. no '*'
6620 aa$=a$(i)
6630 SELect ON CODE(aa$)
6640 REMark set ChkSum to zros at start of sentence
6650 =CODE('$'):ChkSum=0
6660 =CODE('*'):
6670 ChkCode$=a$((i+1) TO (i+2))
6680 EXIT i
6690 =REMAINDER :
6700 REMark ^^ is 'bit-wise exclusive OR' Jan Jones p 40
6710 ChkSum=CODE(aa$)^^ChkSum
6720 END SELect
6730 END FOR i
6740 REMark Checksum is data is two hex chars (8 bits)
6750 IF HEX$(ChkSum,8)=ChkCode$ THEN
6760 RETurn 0: REMark Good result
6770 ELSE
6780 REMark Bad result
6790 PRINT#de%,a$;" ChkSum: ";HEX$(ChkSum,8);" Check code: ";ChkCode$
6800 RETurn -1
6810 END IF
6820 END DEFine CheckSum
6830 REMark
*****************************************************************************
*************************************************************
6840 DEFine PROCedure DisLine(c%,t$,l%)
6850 REMark Displays line padded to l% chars with spaces ..
6860 REMark .. in channel #c%, raw data window, the spaces are needed
6870 REMark when short lines follow long
6880 REMark Had no success with cls so far
6890 PRINT#c%;t$;
6900 tl%=LEN(t$)
6910 IF tl%<l% THEN
6920 PRINT#c%;FILL$(" ",l%-tl%)
6930 ELSE PRINT #c%
6940 END IF
6950 END DEFine DisLine
6960 REMark
```

```
********************************************************************************
************************************************************
6970 DEFine PROCedure Extract_gpgga_data
6980 REMark test line:CLS:PRINT gpsfix$
6990 gpsfix$=chop$(gpsfix$,1)
7000 Time1$=field$(gpsfix$)
7010 gpsfix$=chop$(gpsfix$,1)
7020 Latitude$=field$(gpsfix$)
7030 gpsfix$=chop$(gpsfix$,1)
7040 NSIndicator$=field$(gpsfix$)
7050 gpsfix$=chop$(gpsfix$,1)
7060 Longitude$=field$(gpsfix$)
7070 gpsfix$=chop$(gpsfix$,1)
7080 EWIndicator$=field$(gpsfix$)
7090 gpsfix$=chop$(gpsfix$,1)
7100 PosFix1$=field$(gpsfix$)
7110 gpsfix$=chop$(gpsfix$,1)
7120 Sats$=field$(gpsfix$)
7130 gpsfix$=chop$(gpsfix$,1)
7140 HDP$=field$(gpsfix$)
7150 gpsfix$=chop$(gpsfix$,1)
7160 Altitude$=field$(gpsfix$)
7170 gpsfix$=chop$(gpsfix$,1)
7180 Meter1$=field$(gpsfix$)
7190 gpsfix$=chop$(gpsfix$,1)
7200 Geoid$=field$(gpsfix$)
7210 gpsfix$=chop$(gpsfix$,1)
7220 Meter2$=field$(gpsfix$)
7230 gpsfix$=chop$(gpsfix$,1)
7240 AoD$=field$(gpsfix$)
7250 REMark Test Line:PRINT Time$:PRINT Latitude$:PRINT NSIndicator$:PRINT
Longitude$:PRINT EWIndicator$:PRINT PosFix1$:PRINT Sats$:PRINT HDP$:PRINT
Altitude$:PRINT Meter1$:PRINT Geoid$:PRINT Meter2$:PRINT AoD$:CLOSE:STOP
7260 END DEFine Extract_gpgga_data
7270 REMark
********************************************************************************
************************************************************
7280 DEFine PROCedure Extract_GPGSA_data
7290 REMark Test Line:CLS:PRINT:PRINT gpgsa$
7300 gpgsa$=chop$(gpgsa$,1)
7310 mode2$=field$(gpgsa$)
7320 gpgsa$=chop$(gpgsa$,1)
7330 Mode3$=field$(gpgsa$)
7340 gpgsa$=chop$(gpgsa$,1)
7350 SU1$=field$(gpgsa$)
7360 gpgsa$=chop$(gpgsa$,1)
7370 SU2$=field$(gpgsa$)
7380 gpgsa$=chop$(gpgsa$,1)
7390 SU3$=field$(gpgsa$)
7400 gpgsa$=chop$(gpgsa$,1)
7410 SU4$=field$(gpgsa$)
7420 gpgsa$=chop$(gpgsa$,1)
```

```
7430 SU5$=field$(gpgsa$)
7440 gpgsa$=chop$(gpgsa$,1)
7450 SU6$=field$(gpgsa$)
7460 gpgsa$=chop$(gpgsa$,1)
7470 SU7$=field$(gpgsa$)
7480 gpgsa$=chop$(gpgsa$,1)
7490 SU8$=field$(gpgsa$)
7500 gpgsa$=chop$(gpgsa$,1)
7510 SU9$=field$(gpgsa$)
7520 gpgsa$=chop$(gpgsa$,1)
7530 SU10$=field$(gpgsa$)
7540 gpgsa$=chop$(gpgsa$,1)
7550 SU11$=field$(gpgsa$)
7560 gpgsa$=chop$(gpgsa$,1)
7570 SU12$=field$(gpgsa$)
7580 gpgsa$=chop$(gpgsa$,1)
7590 PDOP$=field$(gpgsa$)
7600 gpgsa$=chop$(gpgsa$,1)
7610 HDOP$=field$(gpgsa$)
7620 gpgsa$=chop$(gpgsa$,1)
7630 VDOP$=field$(gpgsa$)
7640 REMark Test Line:PRINT mode2$:PRINT Mode3$:PRINT SU1$:PRINT SU2$:PRINT
SU3$:PRINT SU4$:PRINT SU5$:PRINT SU6$:PRINT SU7$:PRINT SU8$:PRINT SU9$:PRINT
SU10$:PRINT SU11$:PRINT SU12$:PRINT PDOP$:PRINT HDOP$:PRINT
VDOP$:CLOSE:STOP
7650 END DEFine Extract_GPGSA_data
7660 REMark
*********************************************************************************
*****************************************************************
7670 DEFine PROCedure Extract_GPSRMC_data
7680 REMark Test LineCLS:PRINT gpsrmc$
7690 gpsrmc$=chop$(gpsrmc$,1)
7700 Time2$=field$(gpsrmc$)
7710 gpsrmc$=chop$(gpsrmc$,1)
7720 Status$=field$(gpsrmc$)
7730 gpsrmc$=chop$(gpsrmc$,1)
7740 Latitude2$=field$(gpsrmc$)
7750 gpsrmc$=chop$(gpsrmc$,1)
7760 NSindicator2$=field$(gpsrmc$)
7770 gpsrmc$=chop$(gpsrmc$,1)
7780 Longitude2$=field$(gpsrmc$)
7790 gpsrmc$=chop$(gpsrmc$,1)
7800 EWIndicator2$=field$(gpsrmc$)
7810 gpsrmc$=chop$(gpsrmc$,1)
7820 SOG$=field$(gpsrmc$)
7830 gpsrmc$=chop$(gpsrmc$,1)
7840 COG$=field$(gpsrmc$)
7850 gpsrmc$=chop$(gpsrmc$,1)
7860 SatDate$=field$(gpsrmc$)
7870 REMark Test Line:PRINT Time2$:PRINT Status$:PRINT Latitude2$:PRINT
NSindicator2$:PRINT Longitude2$:PRINT EWIndicator2$:PRINT SOG$:PRINT COG$:PRINT
SatDate$:CLOSE:STOP
```

```
7880 END DEFine Extract_GPSRMC_data
7890 REMark
*************************************************************************************
*****************************************************************
7900 DEFine PROCedure Extract_GPVTG_data
7910 REMark test line:CLS:PRINT gpvtg$
7920 gpvtg$=chop$(gpvtg$,1)
7930 Courset$=field$(gpvtg$)
7940 gpvtg$=chop$(gpvtg$,1)
7950 Ref1$=field$(gpvtg$)
7960 gpvtg$=chop$(gpvtg$,1)
7970 Coursem$=field$(gpvtg$)
7980 gpvtg$=chop$(gpvtg$,1)
7990 Ref2$=field$(gpvtg$)
8000 gpvtg$=chop$(gpvtg$,1)
8010 Speed1$=field$(gpvtg$)
8020 gpvtg$=chop$(gpvtg$,1)
8030 Unit1$=field$(gpvtg$)
8040 gpvtg$=chop$(gpvtg$,1)
8050 Speed2$=field$(gpvtg$)
8060 gpvtg$=chop$(gpvtg$,1)
8070 Unit2$=field$(gpvtg$)
8080 gpvtg$=chop$(gpvtg$,1)
8090 Mode3$=field$(gpvtg$)
8100 REMark test line:PRINT Courset$:PRINT Ref1$:PRINT Coursem$:PRINT Ref2$:PRINT
Speed1$:PRINT Unit1$:PRINT Speed2$:PRINT Unit2$:PRINT Mode3$:CLOSE:STOP
8110 END DEFine Extract_GPVTG_data
8120 REMark
*************************************************************************************
*****************************************************************
8130 DEFine PROCedure Extract_GPGSV1_data
8140 REMark Test Line:CLS:PRINT satlist$(1)
8150 satlist$(1)=chop$(satlist$(1),1)
8160 Nummes1$=field$(satlist$(1))
8170 satlist$(1)=chop$(satlist$(1),1)
8180 Mesnum1$=field$(satlist$(1))
8190 satlist$(1)=chop$(satlist$(1),1)
8200 numsatview1$=field$(satlist$(1))
8210 satlist$(1)=chop$(satlist$(1),1)
8220 satv1$=field$(satlist$(1))
8230 satlist$(1)=chop$(satlist$(1),1)
8240 Sate1$=field$(satlist$(1))
8250 satlist$(1)=chop$(satlist$(1),1)
8260 sata1$=field$(satlist$(1))
8270 satlist$(1)=chop$(satlist$(1),1)
8280 snr1$=field$(satlist$(1))
8290 satlist$(1)=chop$(satlist$(1),1)
8300
 satv2$=field$(satlist$(1))
8310 satlist$(1)=chop$(satlist$(1),1)
8320 sate2$=field$(satlist$(1))
8330 satlist$(1)=chop$(satlist$(1),1)
```

```
8340 sata2$=field$(satlist$(1))
8350 satlist$(1)=chop$(satlist$(1),1)
8360 snr2$=field$(satlist$(1))
8370 satlist$(1)=chop$(satlist$(1),1)
8380 satv3$=field$(satlist$(1))
8390 satlist$(1)=chop$(satlist$(1),1)
8400 sate3$=field$(satlist$(1))
8410 satlist$(1)=chop$(satlist$(1),1)
8420 sata3$=field$(satlist$(1))
8430 satlist$(1)=chop$(satlist$(1),1)
8440 snr3$=field$(satlist$(1))
8450 satlist$(1)=chop$(satlist$(1),1)
8460 satv4$=field$(satlist$(1))
8470 satlist$(1)=chop$(satlist$(1),1)
8480 sate4$=field$(satlist$(1))
8490 satlist$(1)=chop$(satlist$(1),1)
8500 sata4$=field$(satlist$(1))
8510 satlist$(1)=chop$(satlist$(1),1)
8520 snr4$=field$(satlist$(1))
8530 REMark Test Line:PRINT Nummes1$:PRINT Mesnum1$:PRINT numsatview1$:PRINT
satv1$:PRINT Sate1$:PRINT sata1$:PRINT snr1$:PRINT satv2$:PRINT sate2$:PRINT
sata2$:PRINT snr2$:PRINT satv3$:PRINT sate3$:PRINT sata3$:PRINT snr3$:PRINT
satv4$:PRINT sate4$:PRINT sata4$:PRINT snr4$:CLOSE:STOP
8540 END DEFine Extract_GPGSV1_data
8550 REMark
********************************************************************************
************************************************************
8560 DEFine PROCedure Extract_GPGSV2_data
8570 REMark test line:AT 54,0:PRINT satlist$(2)
8580 satlist$(2)=chop$(satlist$(2),1)
8590 Nummes2$=field$(satlist$(2))
8600 satlist$(2)=chop$(satlist$(2),1)
8610 Mesnum2$=field$(satlist$(2))
8620 satlist$(2)=chop$(satlist$(2),1)
8630 numsatview2$=field$(satlist$(2))
8640 satlist$(2)=chop$(satlist$(2),1)
8650 satv5$=field$(satlist$(2))
8660 satlist$(2)=chop$(satlist$(2),1)
8670 Sate5$=field$(satlist$(2))
8680 satlist$(2)=chop$(satlist$(2),1)
8690 sata5$=field$(satlist$(2))
8700 satlist$(2)=chop$(satlist$(2),1)
8710 snr5$=field$(satlist$(2))
8720 satlist$(2)=chop$(satlist$(2),1)
8730 satv6$=field$(satlist$(2))
8740 satlist$(2)=chop$(satlist$(2),1)
8750 sate6$=field$(satlist$(2))
8760 satlist$(2)=chop$(satlist$(2),1)
8770 sata6$=field$(satlist$(2))
8780 satlist$(2)=chop$(satlist$(2),1)
8790 snr6$=field$(satlist$(2))
8800 satlist$(2)=chop$(satlist$(2),1)
```

```
8810 satv7$=field$(satlist$(2))
8820 satlist$(2)=chop$(satlist$(2),1)
8830 sate7$=field$(satlist$(2))
8840 satlist$(2)=chop$(satlist$(2),1)
8850 sata7$=field$(satlist$(2))
8860 satlist$(2)=chop$(satlist$(2),1)
8870 snr7$=field$(satlist$(2))
8880 satlist$(2)=chop$(satlist$(2),1)
8890 satv8$=field$(satlist$(2))
8900 satlist$(2)=chop$(satlist$(2),1)
8910 sate8$=field$(satlist$(2))
8920 satlist$(2)=chop$(satlist$(2),1)
8930 sata8$=field$(satlist$(2))
8940 satlist$(2)=chop$(satlist$(2),1)
8950 snr8$=field$(satlist$(2))
8960 REMark test line: AT 55,0:PRINT Nummes2$;" ";Mesnum2$;" ";numsatview2$;" ";satv5$;"
";Sate5$;" ";sata5$;" ";snr5$;" ";satv6$;" ";sate6$;" ";sata6$;" ";snr6$;" ";satv7$;" ";sate7$;"
";sata7$;" ";snr7$;" ";satv8$;" ";sate8$;" ";sata8$;" ";snr8$:REMark CLOSE:STOP
8970 END DEFine Extract_GPGSV2_data
8980 REMark
**********************************************************************************
*****************************************************************
8990 DEFine PROCedure Extract_GPGSV3_data
9000 REMark Test Line:CLS:PRINT satlist$(3)
9010 satlist$(3)=chop$(satlist$(3),1)
9020 Nummes3$=field$(satlist$(3))
9030 satlist$(3)=chop$(satlist$(3),1)
9040 Mesnum3$=field$(satlist$(3))
9050 satlist$(3)=chop$(satlist$(3),1)
9060 numsatview3$=field$(satlist$(3))
9070 satlist$(3)=chop$(satlist$(3),1)
9080 satv9$=field$(satlist$(3))
9090 satlist$(3)=chop$(satlist$(3),1)
9100 Sate9$=field$(satlist$(3))
9110 satlist$(3)=chop$(satlist$(3),1)
9120 sata9$=field$(satlist$(3))
9130 satlist$(3)=chop$(satlist$(3),1)
9140 snr9$=field$(satlist$(3))
9150 satlist$(3)=chop$(satlist$(3),1)
9160 satv10$=field$(satlist$(3))
9170 satlist$(3)=chop$(satlist$(3),1)
9180 sate10$=field$(satlist$(3))
9190 satlist$(3)=chop$(satlist$(3),1)
9200 sata10$=field$(satlist$(3))
9210 satlist$(3)=chop$(satlist$(3),1)
9220 snr10$=field$(satlist$(3))
9230 satlist$(3)=chop$(satlist$(3),1)
9240 satv11$=field$(satlist$(3))
9250 satlist$(3)=chop$(satlist$(3),1)
9260 sate11$=field$(satlist$(3))
9270 satlist$(3)=chop$(satlist$(3),1)
9280 sata11$=field$(satlist$(3))
```

```
9290 satlist$(3)=chop$(satlist$(3),1)
9300 snr11$=field$(satlist$(3))
9310 satlist$(3)=chop$(satlist$(3),1)
9320 satv12$=field$(satlist$(3))
9330 satlist$(3)=chop$(satlist$(3),1)
9340 sate12$=field$(satlist$(3))
9350 satlist$(3)=chop$(satlist$(3),1)
9360 sata12$=field$(satlist$(3))
9370 satlist$(3)=chop$(satlist$(3),1)
9380 snr12$=field$(satlist$(3))
9390 REMark Test Line:PRINT Nummes3$:PRINT Mesnum3$:PRINT numsatview3$:PRINT
satv9$:PRINT Sate9$:PRINT sata9$:PRINT snr9$:PRINT satv10$:PRINT sate10$:PRINT
sata10$:PRINT snr10$:PRINT satv11$:PRINT sate11$:PRINT sata11$:PRINT snr11$:PRINT
satv12$:PRINT sate12$:PRINT sata12$:PRINT snr12$:CLOSE:STOP
9400 END DEFine Extract_GPGSV3_data
9410 REMark
********************************************************************************
************************************************************
9420 DEFine PROCedure display_data
9430 AT#dd%,0,0:AT#ds%;0,0:INK#ds%;0:REMark To keep the display tidy
9440 REMark Now have all the data needed for display
9450 :
9460 REMark Extract satellites used into satsusd arrary
9470 REMark from $GPGSA line
9480 i=0:REMark Count for sused REPeat loop
9490 :
9500 REMark Process and format time and validity
9510 UTC$=GPSTime$(Time2$)
9520 Inv%=0:REMark to record invalid time and data
9530 IF Status$="V":Inv%=1
9540 :
9550 REMark Display sat data from GPGSA data
9560 PRINT#ds%;\" Sats used   : ";
9570 IF SU1$<>"" THEN satsusd(1)=SU1$:ELSE satsusd(1)=-1
9580 IF SU2$<>"" THEN satsusd(2)=SU2$:ELSE satsusd(2)=-1
9590 IF SU3$<>"" THEN satsusd(3)=SU3$:ELSE satsusd(3)=-1
9600 IF SU4$<>"" THEN satsusd(4)=SU4$:ELSE satsusd(4)=-1
9610 IF SU5$<>"" THEN satsusd(5)=SU5$:ELSE satsusd(5)=-1
9620 IF SU6$<>"" THEN satsusd(6)=SU6$:ELSE satsusd(6)=-1
9630 IF SU7$<>"" THEN satsusd(7)=SU7$:ELSE satsusd(7)=-1
9640 IF SU8$<>"" THEN satsusd(8)=SU8$:ELSE satsusd(8)=-1
9650 IF SU9$<>"" THEN satsusd(9)=SU9$:ELSE satsusd(9)=-1
9660 IF SU10$<>"" THEN satsusd(10)=SU10$:ELSE satsusd(10)=-1
9670 IF SU11$<>"" THEN satsusd(11)=SU11$:ELSE satsusd(11)=-1
9680 IF SU12$<>"" THEN satsusd(12)=SU12$:ELSE satsusd(12)=-1
9690 FOR i=1 TO 12
9700 IF satsusd(i)=-1 THEN EXIT i
9710 IF satsusd(i)<10 THEN PRINT#ds%;"0";
9720 PRINT#ds%; satsusd(i);" ";
9730 END FOR i
9740 :
9750 REMark Spaces at end of 'used' line for shorter overwrite
```

```
9760 PRINT#ds%;"      "
9770 :
9780 PRINT#ds%;" Sats in View: ";
9790 :
9800 REMark display sat signal data
9810 AT#dn%;31,5
9820 REMark reformat signal data for bargraph routine
9830
v$=" ,"&snr1$&","&snr2$&","&snr3$&","&snr4$&","&snr5$&","&snr6$&","&snr7$&","&snr8$
&","&snr9$&","&snr10$&","&snr11$&","&snr12$
9840 FOR i=1 TO 12
9850 v$=chop$(v$,1)
9860 sn$=field$(v$)
9870 IF LEN(sn$)=0 THEN sn$="0"
9880 IF i=1 THEN AT#dn%;3,5
9890 IF i>1 THEN AT#dn%;3,(4*i)+1
9900 IF sn$<10 THEN PRINT#dn%;"0";
9910 PRINT#dn%;sn$!;
9920 sn%=sn$*(255/50):REMark bargraph scaling
9930 holdtime%=3:REMark adjust to lengthen or shorten the bargraph peek hold bar time
9940 IF sn%>peak%(i) THEN peak%(i)=sn%:pht%(i)=holdtime%
9950 IF sn%<peak%(i) THEN pht%(i)=pht%(i)-1
9960 IF sn%=peak%(i) THEN pht%(i)=holdtime%
9970 IF pht%(i)<=0 THEN peak%(i)=sn%:pht%(i)=holdtime%
9980 REMark bargraph ,bar width, bar height, x position, y postion, first/lowest bar colour, second
bar colour, third bar colour, peak hold colour,bargraph level data, threshold from first to second,
threshold from second to third, bar background colour, peak value
9990 bargraph dn%,5,255,(i*24)+10,45,2,0,0,1,sn%,255,255,0,peak%(i)
10000 END FOR i
10010 :
10020 REMark reformat extracted data so it can be read for the FOR/NEXT loop that follows
10030
v$=" ,"&satv1$&","&Sate1$&","&sata1$&","&snr1$&","&satv2$&","&sate2$&","&sata2$&","&
snr2$&","&satv3$&","&sate3$&","&sata3$&","&snr3$&","&satv4$&","&sate4$&","&sata4$&","
&snr4$&","&satv5$&","&Sate5$&","&sata5$&","&snr5$&","&satv6$&","&sate6$&","&sata6$&
","&snr6$&","&satv7$&","&sate7$&","&sata7$&","&snr7$&","&satv8$&","&sate8$&","&sata8$
&","&snr8$&","&satv9$&","&Sate9$&","&sata9$&","&snr9$&","&satv10$&","&sate10$&","&s
ata10$&","&snr10$&","&satv11$&","&sate11$&","&sata11$&","&snr11$&","&satv12$&","&sat
e12$&","&sata12$&","&snr12$
10040 FOR i=1 TO 12
10050 v$=chop$(v$,1)
10060 id$=field$(v$):REMark Satellite Identifier
10070 IF LEN(id$)=0 THEN id$="0"
10080 id%=id$
10090 IF i=1 THEN AT#dn%;31,5
10100 IF i>1 THEN AT#dn%;31,(4*i)+1
10110 IF id$<10 THEN PRINT#ds%;"0";
10120 IF id$<10 THEN PRINT#dn%;"0";
10130 PRINT#ds%;id%!;
10140 PRINT#dn%;id%;
10150 v$=chop$(v$,1):REMark Satellite elevation
10160 el$=field$(v$)
```

```
10170 IF LEN(el$)=0 THEN el$="0"
10180 el%=el$
10190 v$=chop$(v$,1):REMark Satellite azimuth
10200 az$=field$(v$)
10210 IF LEN(az$)=0 THEN az$="0"
10220 az%=az$
10230 v$=chop$(v$,1):REMark Signal from Satellite
10240 sn$=field$(v$)
10250 IF az$="0" OR el$="0" THEN NEXT i
10260 j=0:tint%=seentint%
10270 REPeat chkid
10280 IF satsusd(j)=-1 THEN EXIT chkid
10290 IF satsusd(j)=id% THEN
10300 tint%=usdtint%
10310 END IF
10320 j=j+1
10330 END REPeat chkid
10340 :
10350 IF ShowTrack%=0 THEN
10360 REMark At last can plot satellite in position
10370 REMark Plot only valid data and sat ID's over 0
10380 IF Inv%=0 AND id%>0 THEN :spot el%,az%,id%,tint%
10390 END IF
10400 END FOR i
10410 :
10420 REMark Spaces as end of 'ids' line for shorter overwrite
10430 PRINT#ds%;"      "
10440 :
10450 REMark Print validity
10460 PRINT#ds%;\"   ";
10470 IF Inv%=1:PRINT#ds%;"Inv";: ELSE PRINT#ds%;"V";
10480 PRINT#ds%;"alid Postion  ";
10490 IF Mode3$=1 THEN PRINT#ds%;"Fix not available "
10500 IF Mode3$=2 THEN PRINT#ds%;"2D postion fix      "
10510 IF Mode3$=3 THEN PRINT#ds%;"3D postion fix      "
10520 REMark Spaces to overwrite longer 'Invaid' and postion
 fix messages
10530 :
10540 REMark uses extracted Lat and Long
10550 REMark Chopped gpsrmc$ also used by Track code
10560 CSIZE#ds%;3,1
10570 AT#ds%;3,0
10580 :
10590 REMark Display Latitude
10600 LatDeg$=Latitude2$(1 TO 2)
10610 LatMin$=Latitude2$(3 TO (LEN(Latitude2$)))
10620 PRINT#ds%;"  ";LatDeg$;CHR$(186);
10630 PRINT#ds%;" ";LatMin$;"'"!NSindicator2$;
10640 REMark Latitude
10650 lat=DecDeg(LatDeg$,LatMin$)
10660 IF NSindicator2$=="S" THEN lat=-lat
10670 :
```

```
10680 REMark Display Longitude
10690 LonDeg$=Longitude2$(1 TO 2)
10700 LonMin$=Longitude2$(3 TO (LEN(Longitude2$)))
10710 PRINT#ds%;"  ";LonDeg$;CHR$(186);
10720 PRINT#ds%;" ";LonMin$;"'"!EWIndicator2$
10730 Lon=DecDeg(LonDeg$,LonMin$)
10740 IF EWIndicator2$=="W" THEN Lon=-Lon
10750 :
10760 REMark Display Altitude
10770 AT#ds%;5,1:CSIZE#ds%;0,0:PRINT#ds%;"Altitude"
10780 AT#ds%;11,3:CSIZE#ds%;3,1:PRINT#ds%;Altitude$;" ";Meter1$;"etres"
10790 PRINT_USING#ds%;"#####.#",Altitude$/.3048:PRINT#ds%;" Feet"
10800 :
10810 REMark display UTC, processed way back
10820 AT#ds%;8,2
10830 PRINT#ds%;UTC$
10840 :
10850 REMark display date, processed way back
10860 AT#ds%;9,4
10870 PRINT#ds%;SatDate$(1 TO 2);"/";SatDate$(3 TO 4);"/";SatDate$(5 TO 6)
10880 :
10890 REMark Display Track data,
10900 AT#ds%;10,2
10910 trak$=COG$
10920 speed$=SOG$
10930 IF NumChk(speed$)<>0 THEN CSIZE#ds%;0,0:NEXT orbits
10940 Speed=speed$*1.150779:REMark Convert Knots to m.p.h.
10950 PRINT#ds%;"Track"!trak$;CHR$(186):REMark Bearing
10960 PRINT_USING#ds%;"     at ##.# m.p.h",Speed
10970 PRINT_USING#ds%;"     at ##.# k.p.h",speed$
10980 Brg=trak$
10990 :
11000 CSIZE#ds%;0,0
11010 :
11020 IF ShowTrack% THEN
11030 REMark Show track as a line: colour shows speed
11040 INK#dc%,Spink%(Speed)
11050 IF FirstPt%==0 THEN
11060 POINT#dc%,Lon*asprat*Mctr,lat
11070 FirstPt%=1
11080 ELSE
11090  LINE#dc% TO Lon*asprat*Mctr,lat
11100 END IF
11110 :
11120 REMark Show Speed and bearing as a line
11130 REMark .. Length and colour for speed
11140 REMark .. bearing as direction of line.
11150 CLS#dt%:INK#dt%,Spink%(Speed)
11160 POINT#dt%,0,0
11170 PENDOWN#dt%
11180 TURNTO#dt%,90-Brg:MOVE#dt%,Speed
11190 PENUP#dt%
```

```
11200 END IF
11210 :
11220 REMark Pause to slow down rate of refreshment
11230 PAUSE 50*delay
11240 :
11250 REMark End of repeat loop for continous of display
11260 END REPeat orbits
11270 CLOSE#cs%:IF dtof<>0 THEN CLOSE#dc%
11280 END DEFine display_data
11290 REMark
*********************************************************************************
*****************************************************
11300 DEFine FuNction chop$(str$,skip)
11310 REMark Chps off skip firlds from the start of str$
11320 REMark Haven't had to chop as far as the '*' yet
11330 LOCal i,j
11340 IF (skip < 1) THEN RETurn str$
11350 FOR i=1 TO skip
11360 j="," INSTR str$
11370 str$=str$(j+1 TO)
11380 END FOR
11390 RETurn str$
11400 END DEFine chop$
11410 REMark
*********************************************************************************
***********************************************************
11420 DEFine FuNction NumChk(a$)
11430 REMark Check if a$ contains a decimal number fixed point
11440 REMark Return: 0=ok; 1=empty; 2=>1 d.ps;3=non-numeric
11450 LOCal i%,a%,NoDecPts%:NoDecPts%=0
11460 REMark NoDecPts% hold the numer of dec pts found
11470 IF LEN(a$)=0 THEN RETurn 1
11480 FOR i%=1 TO LEN(a$)
11490 a%=CODE(a$(i%))
11500 REMark 48 and 57 are codes for '0' and '9'
11510 REMark .. I just like < more than <=
11520 IF a%>47 AND a%<58 THEN
11530 NEXT i%
11540 ELSE
11550 REMark Check for dec pt
11560 IF a%=46 THEN
11570 NoDecPts%=NoDecPts%+1:REMark Allow one dec. pt.
11580 IF NoDecPts%>1 THEN RETurn 2:ELSE NEXT i%
11590 END IF
11600 RETurn 3
11610 END IF
11620 END FOR i%
11630 RETurn 0
11640 END DEFine NumChk
11650 REMark
*********************************************************************************
***********************************************************
```

```
11660 DEFine FuNction GPSTime$(t$)
11670 REMark Extracts time from t$ -- copy of RMC input
11680 REMark Should have used chop$ ect, but by this time the
11690 REMark data format seemed stable enough and CBB took over.
11700 t$=chop$(t$,1):REMark Remove '$GPRMC'
11710 RETurn '@ '&t$(1 TO 2)&':'&t$(3 TO 4)&':'&t$(5 TO 6)&'UTC'
11720 END DEFine GPSTime$
11730 REMark
*****************************************************************************
*****************************************************************
11740 DEFine FuNction DecDeg(D$,M$)
11750 IF LEN(D$)==0 OR LEN(M$)==0 THEN RETurn 361
11760 IF ',' INSTR(D$&M$) THEN RETurn 362
11770 RETurn D$+(M$/60)
11780 IF LEN(D$)==0 OR LEN(M$)==0 THEN RETurn 361
11790 END DEFine DecDeg
11800 REMark
*****************************************************************************
*************************************************************
11810 DEFine FuNction field$(str$)
11820 REMark Extracts the first field from the NMEA message data
11830 REMark after it has been chopped to the start of the
11840 REMark field, with comma separated fields so making
11850 REMark no assumption about the field lenght
11860 LOCal k%
11870 REMark last field terminated by* at start of checksum
11880 REMark so need to check for ',' -- if none then '*'
11890 k%="," INSTR str$
11900 IF k%=0 THEN k%="*" INSTR str$
11910 RETurn str$(TO(k%-1))
11920 REMark NOTE error such as empty string must be
11930 REMark dealt with on return from call
11940 END DEFine field$
11950 REMark
*****************************************************************************
*************************************************************
11960 DEFine PROCedure spot(elvn,azm,id,tint)
11970 REMark Draw a blob in tint at elvnm azm and show id
11980 REMark For a polar plot need to convert to x,y coords
11990 REMark Code for line plot added later
12000 LOCal x,y,srad
12010 REMark ignore data if id outside possiable range ..
12020 REMark .. it has happened, usually during start up of RX
12030 IF id>maxid THEN RETurn
12040 srad=(90-elvn)/90:REMark zero to one on plot
12050 x=srad*SIN(azm*PI/180)*asprat
12060 y=srad*COS(azm*PI/180)
12070 REMark Copy previous posn as 'old'
12080 posns(id,0)=posns(id,2)
12090 posns(id,1)=posns(id,3)
12100 REMark Save new posn for 'old' next time
12110 posns(id,2)=x
```

```
12120 posns(id,3)=y
12130 INK#dc%,tint:STRIP#dc%,tint
12140 REMark Draw sat as blob if first time plotted
12150 IF posns(id,4)<1 OR Blobs%=1 THEN
12160 posns(id,4)=1:REMark Remember as blobbed
12170 REMark Draw a blob
12180 FILL#dc%,1
12190 ELLIPSE#dc%,x,y,6E-2,asprat,0
12200 FILL#dc%,0
12210 REMark Show sat id in contrasting ink
12220 IF tint=seetint% THEN INK#dc%,0:ELSE INK#dc%,7
12230 CURSOR#dc%,x,y,-6,-5
12240 IF id<10 THEN PRINT#dc%,0;:REMark Add leading zero?
12250 PRINT#dc%,id:REMark At last, print sat id
12260 ELSE
12270 REMark ... otherwise draw line
12280 INK#dc%,tint:LINE#dc%,posns(id,0),posns(id,1) TO posns(id,2),posns(id,3):INK#dc%,0
12290 END IF
12300 END DEFine spot
12310 REMark
********************************************************************************
************************************************************
12320 DEFine FuNction Spink%(S)
12330 REMark Returns a colour according to the Speed S
12340 SELect ON S
12350 ON S=0 TO 9.999:RETurn 0: REMark Black
12360 ON S=10 TO 19.999:RETurn 59: REMark Brown
12370 ON S=20 TO 29.999:RETurn 2: REMark Red
12380 ON S=30 TO 39.999:RETurn 236: REMark Yellow
12390 ON S=40 TO 49.999:RETurn 22: REMark Orange
12400 ON S=50 TO 59.999:RETurn 3: REMark Green
12410 ON S=60 TO 69.999:RETurn 25: REMark Blue
12420 ON S=70 TO 79.999:RETurn 26: REMark Violet
12430 REMark Shocking pink for over 80m.p.h.
12440 ON S=REMAINDER :RETurn 112
12450 END SELect
12460 END DEFine Spink%
12470 REMark
********************************************************************************
************************************************************
12480 DEFine PROCedure bargraph (bc%,bw%,bh%,bx%,by%,col1%,col2%,col3%,col4%,bd
%,th1%,th2%,bg%,pk%)
12490 REMark bargraph parameters
12500 REMark bc%=Screen channel number
12510 REMark bw%=Bar width
12520 REMark bh%=Bar height
12530 REMark bx%=Bar position x
12540 REMark by%=Bar position y
12550 REMark col1%=First/lowest bar colour
12560 REMark col2%=Second/mid bar colour
12570 REMark col3%=Third/top bar colour
12580 REMark col4%=Peak Hold colour
```

```
12590 REMark bd%=Bargraph level data
12600 REMark th1%=Threshold from first to second bar colour
12610 REMark th2%=Threshold from second to third bar colour
12620 REMark bg%=Bar background colour
12630 REMark pk%=Peak value
12640 BLOCK#bc%;bw%,bh%-bd%,bx%,by%,bg%:REMark background bar
12650 bd2%=bd%
12660 IF bd2%>th1% THEN bd2%=th1%
12670 BLOCK#bc%;bw%,bd2%,bx%,by%+(bh%-bd2%),col1%:REMark First bar level
12680 bd3%=bd%-th1%
12690 IF bd%>th2% THEN bd3%=th2%-th1%
12700 IF bd%>th1% THEN BLOCK#bc%;bw%,bd3%,bx%,by%+(bh%-th1%-
bd3%),col2%:REMark second bar level
12710 bd4%=bd%-th2%
12720 IF bd%>th2% THEN BLOCK#bc%;bw%,bd4%,bx%,by%+(bh%-bd%),col3%:REMark
second bar level
12730 BLOCK#bc%;bw%,2,bx%,(by%+(bh%-pk%))-2,col4%:REMark peek hold bar
12740 END DEFine bargraph
32000 DEFine PROCedure update
32010 SAVE win5_gps_QLToday_GPSProg6
32020 PRINT "Update complete"
32030 END DEFine
```