

# Renesas 2003 Contest Entry H3316

## GPS Display

---

### Description:

A simple and effective way to display the info collected by a Garmin 35 GPS Unit (NMEA protocol).

The unit uses the EVB87 evaluation board with BasicMicro interpreter to receive and display the main data provided by the Garmin GPS35 unit. Any other NMEA compatible GPS unit can be used.

Because the display provided with the EVB87 board was too small for this application I replaced it with a 40X4 LCD produced by SII.



```
Lat:33N19.7331 Lon:117W09.7302
UTC:18:00.56 Date:12/08/03 NrSat:11
Course:00035 Speed:000 Altitude:100.9
Status:GPS OK GpsQual:1 Hdop:0.7 NE
```

GPS DATA  
DISPLAY

```
$GPRMC,180116,A,3319.7336,N,11709.7290,W,0.0,0.0,035.8,081203,013.4,E*60
$PGGA,180116,3319.7336,N,11709.7290,W,1,11,0.7,99.3,M,-32.1,M,0.0,0000.0,0000.0,0000.0,0000.0,0000.0
$SDDBT,0.0,M,0.0,0.0
$SDZDA,03,12,08,18,00,56
$SDMTW,0.0,M,0.0,0.0
$SDDBT,0.0,M,0.0,0.0
$SDZDA,03,12,08,18,00,56
$SDMTW,0.0,M,0.0,0.0
```

GPS RAW  
DISPLAY

The main purpose of the GPS Display is as a test tool for the GPS units. It also could serve as navigation tool – Lat, Lon, UTC time, Date, Heading, Speed and Altitude are displayed in real time.

Two options are available at this moment – GPS formatted data and RAW data for troubleshooting purposes. The option selection is done by the two push buttons.

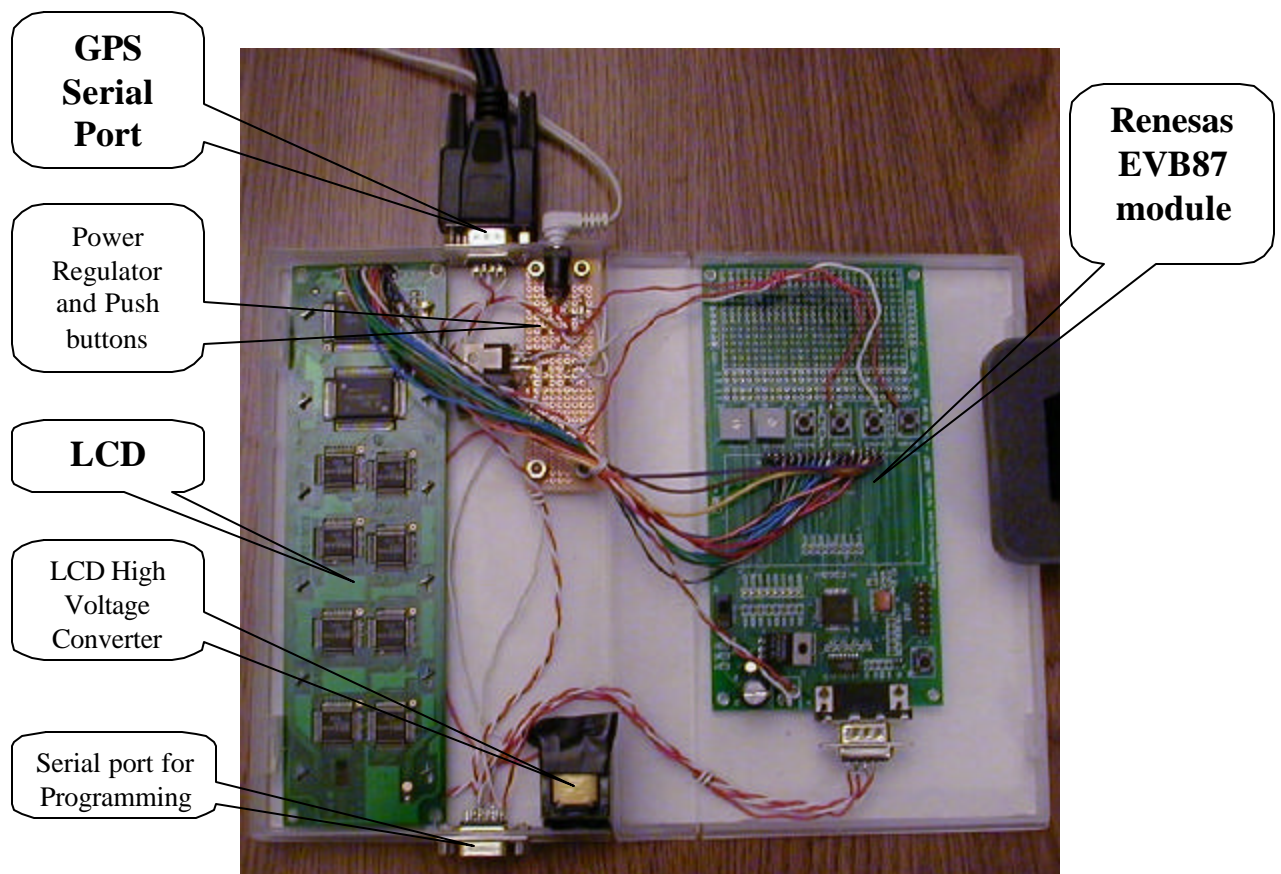
## Hardware:

The hardware of this project is built around the Renesas EVB87 evaluation board. The board uses the H8-3687 CPU that contains 2Kbytes of RAM and 56K Bytes of Flash/program space, a number of general-purpose I/O pins and a RS232 serial level converter.

The LCD was removed and replaced with a SII M4024 9DW 40X4 characters LCD module.

For the Backlight of the LCD a SII High Voltage converter is used.

All these items are packed together with two serial connectors, one power connector and three push buttons in a plastic case.



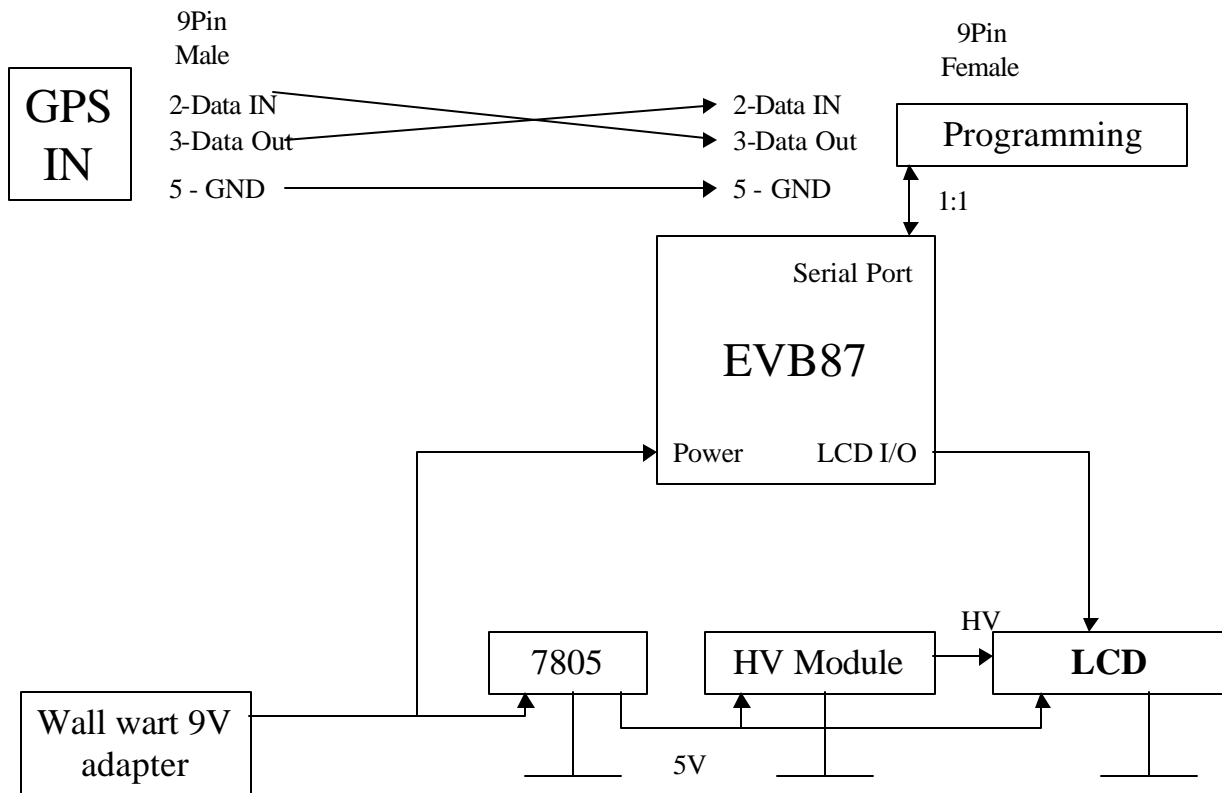
**Power Source:** A 9V/200mA Wall wart adapter is used. The EVB87 module is powered directly from the wall wart. A 7805 5V regulator is used to power the LCD and HV converter (~70V) for the LCD Backlight.

**Serial Connections:** The serial connector from the EVB87 module is replicated 1:1 to the programming connector. The GPS serial input connector uses an X connection to the

EVB87 serial input. The GPS serial in connector and programming connector cannot be used at the same time.

**LCD Connections:** the LCD is SII M4024 9DW – 40X4 characters with Backlight.

LCD Pin		EVB87 Signal
1	D7	P7
2	D6	P6
3	D5	P5
4	D4	P4
5	D3	-
6	D2	-
7	D1	-
8	D0	-
9	Enable	P3
10	R/W	P2
11	RS	P1
12	Vlc	Contrast
13	GND	GND
14	VCC	+5V
15	Enable2	P0





The LCD used has two sections. Each section has two Rows and is initialized by the program. The only one difference between the two sections is the Enable pin – P3 for the first two rows and P0 for the last two rows.

```
; LCD init 40X4, Enable 2 connected to P0  
lcdinit p1 \p3\p7\p6\p5\p4,p2  
lcdwrite p1 \p3\p7\p6\p5\p4,p2,[TWOLINE,CLEAR,HOME,SCR]  
lcdinit p1 \p0\p7\p6\p5\p4,p2  
lcdwrite p1 \p0\p7\p6\p5\p4,p2,[TWOLINE,CLEAR,HOME,SCR]
```

The GPS unit sends multiple data sentences every second. I have selected two of them to decode and print on the LCD screen. The ASCII strings are transmitted over the serial interface 4800baud, 8bit data, 1 bit stop.

Example of real strings sent by the GPS unit:

```
$GPRMC,172443,A,3253.8897,N,11711.4176,W,000.0,031203,013.3,E*66  
172443 = UTC time  
A = GPS ok  
3253.8897,N = Latitude  
11711.4176,W = Longitude  
000.0 = Speed in Knots (*1.15 = miles/hour)  
183.0 = course 0–360  
031203 = Date
```

```
$GPGGA,172842,3253.8896,N,11711.4179,W,1,09,0.9,c,M,-32.9,M,,*70
```

```
172842 = UTC Time  
3253.8896,N = Latitude  
11711.4179,W = Longitude  
1 = GPS Qual  
09 = Nr. of satellites  
0.9 = Hdop  
111.7 = Altitude in meters
```

For more details please see Garmin GPS35 lp TrackPack Technical Specification.  
This is a standard format and any GPS that is NMEA compliant will output similar info.

The program has two options – Raw data and Decoded Data – these options are activated by two push buttons.

The Raw Data Option displays the data as is transmitted by the GPS unit for troubleshooting purposes. The Serin command is used to input data in two strings. The serin command will end if the “\$GPRMC” string is not received in 100 ms or LF is received or the max number of 80 char is received.

The received info is displayed on the LCD with the LCDWRITE command for all the 4 rows ( \$GPRMC received string on rows 1 and 2, \$GPGGA string on rows 3 and 4).

```
line1 var byte(80)  
line2 var byte(80)
```

```

serin S_IN,i4800,100,wline2,[wait("$GPRMC,"),str line1\80\10] ; wait until CR,LF - LF
is discarded or 80 char max
wline2
serin S_IN,i4800,100,main,[wait("$GPGGA,"),str line2\80\10]
wline3
lcdwrite p1 \p3\p7\p6\p5\p4,p2,[SCRRAM,"$GPRMC," ,str line1\80\13] ;type until CR, CR is
discarded
lcdwrite p1 \p0\p7\p6\p5\p4,p2,[SCRRAM,"$GPGGA," ,str line2\80\13]

```

The Decoded Data Option display the data parsed in a easy to read format.  
The sequence for serial data input is similar with the sequence used for Raw Data but in this case each parameter is extracted from the string as necessary.

```

serin S_IN,i4800,100,gpgga,[wait("$GPRMC,"),skip 7, status,skip 26, sdec nrkspeed, skip 2,
sdec nrcourse, skip 2, str date\6]
gpgga
serin S_IN,i4800,100,main,[wait("$GPGGA,"), str utc\6, skip 1, str lat\9,skip 1,lathemi,skip
1,str lon\10,skip 1,lonhemi,skip 1,gpsqual,skip 1,str nrsat\2,skip 1,str hdop\4\'," ,str alti\7\'," ]

```

Once all the parameters are parsed if the status is “GPS OK” then the speed in miles per hour is computed (speed in knots\*1.15) and the heading is translated from the 0-360 degrees reading in a N,NE,NW... format.

```

; convert nrcourse to heading
if nrcourse>22 then goto a1
hhead="N "
goto aend
a1
if nrcourse>67 then goto a2
hhead="NE"
goto aend
a2
if nrcourse>112 then goto a3
hhead="E "
goto aend
a3
if nrcourse>157 then goto a4
hhead="SE"
goto aend
.....

```

All this info is sent to the LCD in four command lines:

```

lcdwrite p1 \p3\p7\p6\p5\p4,p2,[SCRRAM,"Lat:",str lat\2,lathemi,str lat(2)\7," Lon:",str
lon\3,lonhemi,str lon(3)\7]
lcdwrite
p1 \p3\p7\p6\p5\p4,p2,[SCRRAM+$40,"UTC:",utc(0),utc(1),":",utc(2),utc(3),".",utc(4),utc(5),"
Date:",date(2),date(3),"/",str date\2,"/",date(4),date(5)," NrSat:",str nrsat\2]
lcdwrite p1 \p0\p7\p6\p5\p4,p2,[SCRRAM,"Course:",sdec nrcourse\3," Speed:",sdec
nrkspeed\3," Altitude:",str alti\7]
lcdwrite p1 \p0\p7\p6\p5\p4,p2,[SCRRAM+$40,"Status:",str status1\7,"
GpsQual:",gpsqual," Hdop:",str hdop\4," ",str hhead\2]

```

At the end the program control goes back to the main loop in order to check for the selection push-buttons.

## Program Listing:

```
;      GPS DISPALY
;      Renesas 2003 Contest Entry H3316
;
;

work var byte
temp var word
nrkspeed var long      ; temp val for speed
nrcourse var word      ; temp val for course
nrb var long
nrc var long
hhead var byte(2)      ; heding N,NE, etc

;temp1 var byte

flags var byte

flags = 1
; LCD init 40X4, Enable 2 connected to P0
lcdinit p1 \p3\p7\p6\p5\p4,p2
lcdwrite p1 \p3\p7\p6\p5\p4,p2,[TWOLINE,CLEAR,HOME,SCR]
lcdinit p1 \p0\p7\p6\p5\p4,p2
lcdwrite p1 \p0\p7\p6\p5\p4,p2,[TWOLINE,CLEAR,HOME,SCR]

PCRS6 = 0xFF ;Set all the LED
pins as outputs(PCRS is a shadow register that stores the state of PCR since PCR is a write only
register)
PCR6 = 0xFF

temp = 0
goto test2

main
;      button p8,0,0,0,work,1,test1 ;Note that usually you need to
use different work variables for each button command, but since I'm not using delay or repeat rates I
can use the same variable
      button p9,0,0,0,work,1,test2
      button p10,0,0,0,work,1,test3
;      button p11,0,0,0,work,1,test4
      branch flags,[none,serial,typeraw]
      goto main

none
serial

; define the variables used for gps sentences
```

```

utc var byte(6) ; time
lat var byte(9) ; latitude
lathemi var byte
lon var byte(10) ; longitude
lonhemi var byte
gpsqual var byte ; gps signal quality
nrsat var byte(2); # or satellites
hdop var byte(4); horizontal dilution of precision
alti var byte(7) ; altitude
status var byte ; gps fix status A=OK
kspeed var byte(5) ; speed in knots * 1.15 = miles/h
course var byte(5)
date var byte(6)
status1 var byte(7) ; gps status result

; ascii , = dec 44
hdop = " "
alti = " "
serin S_IN,i4800,100,gpgga,[wait("$GPRMC,"),skip 7, status,skip 26, sdec nrkspeed, skip 2,
sdec nrcourse, skip 2, str date\6]
gpgga
serin S_IN,i4800,100,main,[wait("$GPGGA,"), str utc\6, skip 1, str lat\9,skip 1,lathemi,skip
1,str lon\10,skip 1,lonhemi,skip 1,gpsqual,skip 1,str nrsat\2,skip 1,str hdop\4\'," ,str alti\7\'," ]

if status="A" then
    status1="GPS OK "
    ; compute kspeed = kspeed*1.15 - convert to miles
    nrb = 115 ; 1.15 *100...for int math
    nrc = nrkspeed*nrb
    nrkspeed = nrc / 100 ; go back to normal

    ; convert nrcourse to heading

        if nrcourse>22 then goto a1
            hhead="N "
            goto aend
a1
        if nrcourse>67 then goto a2
            hhead="NE"
            goto aend
a2
        if nrcourse>112 then goto a3
            hhead="E "
            goto aend
a3
        if nrcourse>157 then goto a4
            hhead="SE"
            goto aend
a4
        if nrcourse>202 then goto a5
            hhead="SW"
            goto aend
a5
        if nrcourse>247 then goto a6
            hhead="W "
            goto aend

```

```

a6
        if nrcourse>292 then goto a7
            hhead="NW"
            goto aend
a7
        hhead="N "
aend
    else
        status1="GPS BAD"
        hdop = "----"
        alti = "-----"
        nrkspeed = 0
        hhead = "--"
    endif

        lcdwrite p1 \p3\p7\p6\p5\p4,p2,[SCRRAM,"Lat:",str lat\2,lathemi,str lat(2)\7," Lon:",str
lon\3,lonhemi,str lon(3)\7]
        lcdwrite
p1 \p3\p7\p6\p5\p4,p2,[SCRRAM+$40,"UTC:",utc(0),utc(1),":",utc(2),utc(3),".",utc(4),utc(5),"
Date:",date(2),date(3),"/",str date\2,"/",date(4),date(5)," NrSat:",str nrsat\2]
        lcdwrite p1 \p0\p7\p6\p5\p4,p2,[SCRRAM,"Course:",sdec nrcourse\3," Speed:",sdec
nrkspeed\3," Altitude:",str alti\7]
        lcdwrite p1 \p0\p7\p6\p5\p4,p2,[SCRRAM+$40,"Status:",str status1\7,"
GpsQual:",gpsqual," Hdop:",str hdop\4," ",str hhead\2]
        goto main

typeraw
; type raw data to the display

line1 var byte(80)
line2 var byte(80)
        serin S_IN,i4800,100,wline2,[wait("$GPRMC,") ,str line1\80\10] ; wait until CR,LF - LF
is discarded or 80 char max
wline2
        serin S_IN,i4800,100,main,[wait("$GPGGA,") ,str line2\80\10]
wline3
        lcdwrite p1 \p3\p7\p6\p5\p4,p2,[SCRRAM,"$GPRMC," ,str line1\80\13] ;type until CR, CR is
discarded
        lcdwrite p1 \p0\p7\p6\p5\p4,p2,[SCRRAM,"$GPGGA," ,str line2\80\13]
        goto main

test2
        lcdwrite p1 \p3\p7\p6\p5\p4,p2,[CLEAR,SCRRAM + $40," GPS DISPLAY V1.0 "]
        lcdwrite p1 \p0\p7\p6\p5\p4,p2,[CLEAR,SCRRAM, " (c)2003 Contest Entry H3316"]
        flags = 1
        goto main

test3
        lcdwrite p1 \p3\p7\p6\p5\p4,p2,[CLEAR,"Raw GPS Data"]
        lcdwrite p1 \p0\p7\p6\p5\p4,p2,[CLEAR]

```

```
flags = 2  
goto main
```

## Sources:

- Garmin GPS35 Ip TrackPack Technical Specification – [www.garmin.com](http://www.garmin.com)
- EVB87 Technical Documentation – [www.renesas.com](http://www.renesas.com)
- LCD controller doc. - [http://www.seiko-usa-eed.com/lcd/products/char\\_mods/](http://www.seiko-usa-eed.com/lcd/products/char_mods/)
- Basic Micro - [www.basicmicro.com](http://www.basicmicro.com)