References:

[1] http://www.aero.org/publications/GPSPRIMER/

[2] http://www.trimble.com

[3] http://www.colorado.edu/geography/gcraft/notes/gps/gps_f.htm

[4] http://hyperphysics.phy-astr.gsu.edu/hbase/gpsrec.html

[5] http://www.auf.asn.au/navigation/gps.html#gps_receivers

[6] http://www.papyrus.co.il/FAQ/dgps_-_differntial_gps_positing.htm

[7] http://www.gpsworld.com/gpsworld/static/staticHtml.

[8] http://www.garmin.com/aboutGPS

[9] Patrick Comilleri, "GPS car navigation and instrumentation panel design," University of Malta, 2000, pp 1-10.

[10] http://www.gpsinformation.net/

[11] Michel O'Connor, Thomas Bell, Gabriel Elkaim, Bradford Parkinson," automatic steering farm vehicles using GPS "Stanford University California, 1996

[12] Marvin May, Pus/Arl, Alison Brawn," application of digital storage receiver for enhanced signal processing ", *NAVSYS* Corporation**,** 1999

[13] Caroline Eriksson &Pierre Heroux,"GPS location for GIS", Toronto, 2001, pp 1-15.

[14] Mark Nylon, Alison Brown & J.Jclark,**"** Kinematic GPS-Inertial Navigation on a Tactical Fighter," *NAVSYS Corporation* Portland, 2003, pp 1-9.

[15] NAVSTAR," Global Positioning System Standard Service Signal Specification ", 1995

[16] Michael S. Braaasch .MEMBER, IEEE and A.J VAN,"GPS Receiver Architecture and Measurement ", VOL, 87, 1999, PP 25-55.

References:

[17] http:// www.unav-micro.com/gsp tutorial signal and data

[18] Graham Smillie,"Analog and Digital Communications Techniques, "London, 1999

[19]http://hyperphysics.phy-astr.gsu.edu/hbase/gpsrec.html

[20] http://www.gpsinformation.org/dale/nmea.htm#hardware

[21] http://vancouver-webpages.com/peter/nmeafaq.txt

[22] http://www.kh-gps.de/nmea-faq.htm

[23] http://www.commlinx.com.au/NMEA_sentences.htm

[24] Myke Predko,"Programming and Customizing Pic®Micro Microcontrollers, "USA, 2001

[25] M. Morris MANO, "Digital Logic and Computer Design," New Delhi, 1994.

[26] Abhishek Patil & Humagun Kabir," Platform Independent "GPS" Receiver, "fall, 2002

[27] http:// www.zarlink.com

[28]Frederick F. Driscoll, Robert F Coughlin and Robert S Villanucci,"Data Acquisition and Process Control with the M68hc11microcontroller," New Jersey, 2000

[29] http:// www.epanorama.net/links/microprocessor.html

[30] http://www .national.com

[31] Albert Paul Malvino, Donald P.Leach," Digital Principle and Applications, "Singapore, 1986

References:

[32] Douglas E. Comer," Computer Networks and Internets," New Jersey, 2001

[33] Christopher E. Strangio,"Data Communications Basics," Massachusetts, 2000

[34] Douglas V. Hall," Microprocessor and Interfacing," Singapore .1986

[35] Behrouz A. Farouzan," Data Communications and Networking," New Delhi, 2003

[36] http://www.ctips.com/rs232.htm

[37] George A. Smith," computer interfacing," Britain, 2000

[38] PeL VAN, "PC Interfacing," UK, 1998

[39] Barry B .Brey," The Intel Microprocessors ", USA, 2003

[40] http://www.strath.ac.uk/IT/Docs/Ccourse/

[41] Clarion software corporation," Top Speed," UK, 1992

APPENDIX (A) Source Code "C language"

Note: download and unzip gps1_5.zip rather than cutting and pasting from below.

```
/*
   FILE: gps1_5.c
   AUTH: P.OH
   DESC: Garmin EMap connected to COM1
   REFS: Uses ibmcom serial libraries
   NOTE: To compile: tcc -ml gps1_5.c ibmcom3.obj
*/

/* Defines required for serial i/o */
#define COM_PORT   1        /* Serial device connected to COM 1
*/
#define SPEED      4800           /* baud rate = 4800 */
#define CR         0x0d
#define LF         0x0a
#define ESC        0x1b
#define BEEP       0x07

/* Some helpful defines */
#define SPACE   0x20
#define COMMA   0x2C
#define MAXSIZE    100            /* GPS at most, sends 80 or so
chars per message string.  So set maximum to 100 */

#include < stdio.h >
#include < ctype.h >              /* required for the isalnum
function */
#include < stdlib.h >
#include < string.h >
#include < conio.h >
#include < math.h >
#include < dos.h >
#include "ibmcom3.h"              /* for serial */

/* Prototypes */
void comm_setting(void);    /* Set com port */
void close_com(void);       /* Close com port */

int main(void) {

  unsigned char  charRead;               /* char read from COM
port */
  unsigned char       stringRead[MAXSIZE];             /* Buffer
collects chars read from GPS */
  unsigned char  tempString[MAXSIZE];
  unsigned char  timeString[12];
  unsigned char  latitudeString[11];
```

```
  unsigned char  latitudeCardinalString[3];
  unsigned char  longitudeString[12];
  unsigned char  longitudeCardinalString[3];

  unsigned char  *pChar;
  unsigned char  dummyChar;

  unsigned  long utcTime, estTime;        /* Coordinated
Universal Time and Eastern Standard Time */
  unsigned  long utcHour, estHour;
  unsigned  long utcMinutes, estMinutes;
  unsigned  long utcSeconds, estSeconds;
  unsigned  char lastCommaPosition;

  float        latitude;
  int          latDegrees;
  float               latMinutes;

  float        longitude;
  int          longDegrees;
  float               longMinutes;

  FILE           *gpsFile;                /* Text file of GPS
strings read */
  unsigned int   j, k;                           /* dummy
variable */
  unsigned int        i;                         /* Number of
chars read per GPS message string */
  unsigned int        numLinesRead;                    /* Number
of GPS strings read */

  dummyChar = 'A'; pChar = &dummyChar;
  gpsFile = fopen("gpsData.txt", "w");

  printf("Initializing port...");
  comm_setting();
  printf("done\n");

  numLinesRead = 0;

  printf("Entering while loop...\n");
  do {
      charRead = com_rx();         /* read char from serial port
*/
      if(charRead == '$') {     /* GPS messages start with $ char
*/
         i = 0;
         numLinesRead++;
```

```
         stringRead[i] = charRead;
         do {
            charRead = com_rx();
            if( (charRead != '\0') && (isalnum(charRead) ||
isspace(charRead) || ispunct(charRead)) ) {
              i++;
              stringRead[i] = charRead;
            }
         } while(charRead != CR);

         /* By this point, a complete GPS string has been read so
save it to file */
         /* Append the null terminator to the string read */
         stringRead[i+1] = '\0';

         /* Analyze string that we collected */
         j = 0;
         pChar = stringRead;
         while(*(pChar+j) != COMMA) {
              tempString[j] = *(pChar+j);
              j++;
         }
         tempString[j] = '\0';

         /* Check if string we collected is the $GPGGA message */
         if(tempString[3] == 'G' && tempString[4] == 'G' &&
tempString[5] == 'A') {
             /*
               Found GPGGA string.  It has 14 commas total.  Its
NMEA sentence structure is:


$GPGAA,hhmmss.ss,ddmm.mmmm,n,dddmm.mmmm,e,q,ss,y.y,a.a,z,g.g,z,t.
t,iii*CC
               |    |    |    |    |    |    |    |    |    |
|    |    |    |    |
               0        1         2         3         4         5
6         7

012345678901234567890123456789012345678901234567890123456789012345
56789012

               where:

               GPGAA        : GPS fixed data identifier
               hhmmss.ss    : Coordinated Universal Time (UTC),
also known as GMT
```

```
              ddmm.mmmm,n  : Latitude in degrees, minutes and
cardinal sign
              dddmm.mmmm,e : Longitude in degrees, minutes and
cardinal sign
              q            : Quality of fix.  1 = there is a fix
              ss           : Number of satellites being used
              y.y          : Horizontal dilution of precision
              a.a,M        : GPS antenna altitude in meters
              g.g,M        : geoidal separation in meters
              t.t          : Age of the defferential correction
data
              iiii         : Deferential station's ID
              *CC          : checksum for the sentence
            */

            pChar = stringRead;

            /* Get UTC time */
            j = 7;  /* start of time field */
            k = 0;
            while(*(pChar+j) != COMMA) {
                timeString[k] = *(pChar+j);
                j++;
                k++;
            }
            lastCommaPosition = j;
            timeString[k] = '\0';
            sscanf(timeString, "%ld", &utcTime);
            utcHour = (utcTime/10000);   /* extract Hours from
long */
            utcMinutes = (utcTime - (utcHour*10000))/100;  /*
extract minutes from long */
            utcSeconds = utcTime - (utcHour*10000) -
(utcMinutes*100); /* extract seconds from long */

            if(utcHour >= 4 && utcHour <= 23) estHour = utcHour
- 4;
             else estHour = utcHour + 20;
            estMinutes = utcMinutes;
            estSeconds = utcSeconds;

            /* NB: %02ld formats long to print 2 chars wide,
padding with 0 if necessary */
            printf("%02ld:%02ld:%02ld UTC = %02ld:%02ld:%02ld
EST", utcHour, utcMinutes, utcSeconds, estHour, estMinutes,
estSeconds);

            /* Get lattitude: ddmm.mmmm */
```

```
            pChar = stringRead;
            j = lastCommaPosition + 1;
            k = 0;
            while(*(pChar+j) != COMMA) {
                latitudeString[k] = *(pChar+j);
                j++;
                k++;
            }
            lastCommaPosition = j;
            latitudeString[k] = '\0';

            sscanf(latitudeString, "%f", &latitude);
            latDegrees = (int)(latitude/100);
            latMinutes = (float)(latitude - latDegrees*100);
            printf("\t%02d DEG\t%2.4f MIN", latDegrees,
latMinutes);

            /* Get lattitude Cardinal direction */
            pChar = stringRead;
            j = lastCommaPosition + 1;
            k = 0;
            while(*(pChar+j) != COMMA) {
                latitudeCardinalString[k] = *(pChar+j);
                j++;
                k++;
            }
            lastCommaPosition = j;
            latitudeCardinalString[k] = '\0';
            printf(" %s", latitudeCardinalString);

            /* Get longitude: dddmm.mmmm */
            pChar = stringRead;
            j = lastCommaPosition + 1;
            k = 0;
            while(*(pChar+j) != COMMA) {
                longitudeString[k] = *(pChar+j);
                j++;
                k++;
            }
            lastCommaPosition = j;
            longitudeString[k] = '\0';

            sscanf(longitudeString, "%f", &longitude);
            longDegrees = (int)(longitude/100);
            longMinutes = (float)(longitude - longDegrees*100);
            printf("\t%03d DEG\t%2.4f MIN", longDegrees,
longMinutes);
```

```
            printf("\n");
         } /* else not a GPGGA sentence */

         fprintf(gpsFile, "%d: (%d) %s\n", numLinesRead, i,
stringRead);

      } /* otherwise not a $ character... so loop back until one
arrives */
  } while(!kbhit());

  printf("Exiting...");
  close_com();   /* Finished with serial port so close it */
  fclose(gpsFile);
  printf("done\n");
  return (0);

} /* end of main */

void comm_setting(void) {

  int  dummy;

  dummy = com_install(COM_PORT);
  if(dummy != 0) {
       switch (dummy)       {
         case 1  : printf("Invaid port number\n");
                   break;
         case 2  : printf("No UART fot specified port\n");
                   break;
         case 3  : printf("Drivers already installed\n");
                   break;
         default : printf("Err #%d\n", dummy);
                   break;
       }
       exit(1);
  } com_raise_dtr();

  com_set_speed(SPEED);
  com_set_parity(COM_NONE, STOP_BIT_1);
}

void close_com(void) {
  com_lower_dtr();
  com_deinstall();
}
```

APPENDIX (B) List of Acronyms

bps	bits per second
BPSK	Bipolar-Phase Shift Key
C/A	Coarse/Acquisition
CS	Control Segment
dBi	Decibels, isotropic
dBw	Decibels, watt
DCE	Data Communication Equipment
DOD	Department of Defense
DOP	Dilution of Precision
DTE	Data Terminal Equipment
GPS	Global Positioning System
HOW	Hand-Over Word
LSB	Least Significant Bit
Mbps	Million bits per second
MCS	Master Control Station
MSB	Most Significant Bit
NMEA	National Marine Electronics Association
OCS	Operational Control System
PRN	Pseudo Random Noise
PVT	Position, Velocity, Time
RF	Radio Frequency
SA	Selective Availability
SS	Space Segment
SV	Space Vehicle
TLM	Telemetry

APPENDIX (B) List of Acronyms

TOW	Time of Week
UE User	Equipment
U.S.	United States
UTC	Universal Coordinated Time
WGS-84	World Geodetic System 1984