# How to interface GPS with 8051 Microcontroller (AT89C51)

The GPS module continuously transmits serial data (RS232 protocol) in the form of sentences according to NMEA standards. The latitude and longitude values of the location are contained in the **GPGGA** sentence (refer NMEA format). In this program, these values are extracted from the GPGGA sentence and are displayed on LCD.

The serial data is taken from the GPS module through MAX232 into the SBUF register of 8051 controller (refer serial interfacing with 8051). The serial data from the GPS receiver is taken by using the Serial Interrupt of the controller. This data consists of a sequence of NMEA sentences from which GPGGA sentence is identified and processed.

The extraction of location values is done as follows. The first six bytes of the data received are compared with the pre-stored ($GPGGA) string and if matched then only data is further accounted for; otherwise the process is repeated again. From the comma delimited GPGGA sentence, latitude and longitude positions are extracted by finding the respective comma positions and extracting the data. The latitude and longitude positions extracted are displayed on theLCD interfaced with AT89C51.

To obtain more details (other than latitude and longitude) from the GPS receiver, GPRMS sentence can be used. Refer next article.

**The circuit connections are as follows:**

Receiver1 ($R_1$) of MAX232 has been used for the serial communication. The receiver pin of GPS module is connected to $R_1$IN (pin13) of MAX232. $R_1$OUT (pin 12) of MAX232 is connected to RxD (P3.0) of AT89C51.

Pins 1-3 of port P1 (P1.0, P1.1 & P1.2 respectively) of AT89C51 are connected to the control pins (RS, R/W& EN) of LCD. The data pins of LCD are connected to Port P2 of the controller. The latitude and longitude positions are displayed on the LCD.

The GPS module continuously transmits serial data (RS232 protocol) in the form of sentences according to NMEA standards. The latitude, longitude, time, date and speed values of the receiver are contained in the **GPRMC** sentence as given in the following example (also refer NMEA format for other sentences). In this project, these values are extracted from the GPRMC sentence and are displayed on LCD.

Example : $GPRMC,132455.970,A,2651.0145,N,07547.7051,E,0.50,342.76,301010,,,A*64

where:

| | |
|---|---|
| RMC | Recommended Minimum sentence C |
| 132455.970 | Fix taken at 13:24:55.970 UTC |
| A | Status A=Active or V=Void. |
| 2651.0145, N | Latitude 26 deg 51.0145' N |
| 07547.7051, E | Longitude 075 deg 47.7051' E |
| 0.50 | Speed over the ground in knots |
| 342.76 | Track angle in degrees True |
| 301010 | Date : 30th of October 2010 |
| Empty field (xxx.x, y) | Magnetic Variation |
| *64 | The checksum data, always begins with * |

The serial data is taken from the GPS module through MAX232 into the SBUF register of 8051 controller (refer serial interfacing with 8051). The serial data from the GPS receiver is taken by using the Serial Interrupt of the controller. This data consists of a sequence of NMEA sentences from which GPRMC sentence is identified and processed.

The extraction of required values is done as follows. The first six bytes of the data received are compared with the pre-stored ($GPRMC) string and if matched then only data is further accounted for; otherwise the process is repeated again. From the comma delimited GPRMC sentence, latitude, longitude, date, time, speed values are extracted by finding the respective comma positions. The values thus extracted are displayed on the LCD interfaced with AT89C51.

**The circuit connections are as follows:**

Receiver1 ($R_1$) of MAX232 has been used for the serial communication. The receiver pin of GPS module is connected to $R_1$IN (pin13) of MAX232. $R_1$OUT (pin 12) of MAX232 is connected to RxD (P3.0) of AT89C51.

Pins 1-3 of port P1 (P1.0, P1.1 & P1.2 respectively) of AT89C51 are connected to the control pins (RS, R/W& EN) of LCD. The data pins of LCD are connected to Port P2 of the controller. The latitude and longitude positions are displayed on the LCD.

```
/* Basic program to show latitude and longitude on LCD extracted from GPGGA
statement */

#include<reg51.h>
#define port2 P2
sbit rs = P1^0;
sbit rw = P1^1;
```

```
sbit e = P1^2;
char info[70];
char test[6]={"$GPGGA"};
char comma_position[15];
unsigned int check=0,i;
unsigned char a;
void receive_data();
void lcd_latitude();
void lcd_longitude();

//DELAY FUNCTION
void delay(unsigned int msec)
{
        int i,j ;
        for(i=0;i<msec;i++)
        for(j=0;j<1275;j++);
}

// LCD COMMAND SENDING FUNCTION
void lcd_cmd(unsigned char item)
{
        port2 = item;
        rs= 0;
        rw=0;
        e=1;
        delay(1);
        e=0;
        return;
}

// LCD DATA SENDING FUNCTION
void lcd_data(unsigned char item)
{
        port2 = item;
        rs= 1;
        rw=0;
        e=1;
        delay(1);
        e=0;
        return;
}

 // LCD STRING SENDING FUNCTION
void lcd_string(unsigned char *str)
{
        int i=0;
        while(str[i]!='\0')
        {
        lcd_data(str[i]);
        i++;
        delay(10);
        }
        return;
}

// SERIAL PORT SETTING
void serial()
{
        TMOD=0x20;      //MODE=2
```

```
        TH1=0xfa;             // 4800 BAUD
        SCON=0x50  ;    // SERIAL MODE 1 ,8- BIT DATA ,1 STOP BIT ,1 START BIT ,
RECEIVING ON
        TR1=1;            //TIMER START
}

void find_comma()
{
        unsigned int i,count=0;
        for(i=0;i<70;i++)
        {
                if(info[i]==',')
                {
                        comma_position[count++]=i;
                }
         }
}
void compare()
{
        IE=0x00;        //Interrupt disable
        find_comma();       //Function to detect position of comma in the string
        lcd_latitude();    //Function to show Latitude
        lcd_longitude();   //Function to show Longitude
        check=0;
        IE=0x90;            //Interrupt enable
}
void receive_data()               interrupt 4
{
        info[check++]=SBUF;         //Read SBUF
        if(check<7)               //Condition to check the required data
        {
                if(info[check-1]!=test[check-1])
                check=0;
        }
        RI=0;
}
void lcd_shape()       //Function to create shape of degree
{
        lcd_cmd(64);
        lcd_data(10);
        lcd_data(17);
        lcd_data(17);
        lcd_data(10);
        lcd_data(0);
        lcd_data(0);
        lcd_data(0);
        lcd_data(0);
}

void lcd_latitude()               //Function to display Latitude
{
        unsigned int c2=comma_position[1]; //Position of second comma
        lcd_shape();
        lcd_cmd(0x01);            // Clear LCD display
        lcd_cmd(0x84);            //Move cursor to position 6 of line 1
        lcd_string("LATITUDE");   //Showing Latitude
        lcd_cmd(0xC0);                    //Beginning of second line
        lcd_data(info[c2+1]);
        lcd_data(info[c2+2]);
```

```
        lcd_data(0);                        //Degree symbol
        lcd_data(info[c2+3]);
        lcd_data(info[c2+4]);
        lcd_data(info[c2+5]);
        lcd_data(info[c2+6]);
        lcd_data(info[c2+7]);
        lcd_data(info[c2+8]);
        lcd_data(info[c2+9]);
        lcd_data(0x27);          //ASCII of minute sign(')
        lcd_data(info[c2+10]);
        lcd_data(info[c2+11]);
        delay(250);
}

void lcd_longitude()
{
        unsigned int c4=comma_position[3];
        lcd_cmd(0x01);               //Clear LCD display
        lcd_cmd(0x84);               //Move cursor to position 4 of line 1
        lcd_string("LONGITUDE");         //Showing Longitude
        lcd_cmd(0xC0);                      //Begining of second line
        lcd_data(info[c4+1]);
        lcd_data(info[c4+2]);
        lcd_data(info[c4+3]);
        lcd_data(0);
        lcd_data(info[c4+4]);
        lcd_data(info[c4+5]);
        lcd_data(info[c4+6]);
        lcd_data(info[c4+7]);
        lcd_data(info[c4+8]);
        lcd_data(info[c4+9]);
        lcd_data(info[c4+10]);
        lcd_data(0x27);               //ASCII of minute sign(')
        lcd_data(info[c4+11]);
        lcd_data(info[c4+12]);
        delay(250);
}
void main()
{
        serial();
        lcd_cmd(0x38);           //2 LINE, 5X7 MATRIX
        lcd_cmd(0x0e);         //DISPLAY ON, CURSOR BLINKING
        IE=0x90;
        while(1)
        {
                if(check==69)
                compare();
        }
}
```