Freescale Semiconductor
Application Note

Document Number: AN4046
Rev. 0, 01/2010

Extracting Latitude and Longitude from a GPS Device for Windows Embedded CE™ 6.0

by *Multimedia Application Division*
Freescale Semiconductor, Inc.
Austin, TX

The following document describes how to extract information from a Global Positioning System (GPS) device, using the GPS library located on the Board Support Package (BSP) for PDK board. This configuration is supported for Windows Embedded CE 6.0.

Windows Embedded CE 6.0 supports GPS controlling interfaces using the Global Position System Intermediate Driver (GPSID). This driver layer works above the Hardware Abstraction Layer (HAL), and hence allows compatibility between the applications and other GPS devices in such a way that any GPS device can be used for the client application.

Contents

freescale™ semiconductor

Figure 1 describes the GPS architecture. The client applications interface with the GPSID API in windows layer, irrespective of the GPS hardware device connected to the port. The GPSID driver has internal sublayers and it is connected directly with the HAL GPS driver.

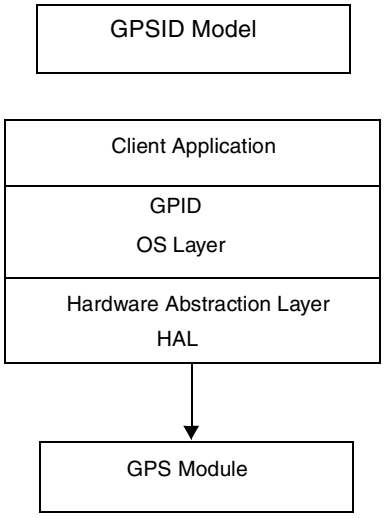

Figure 1. GPS Architecture

The GPSID works with National Marine Electronic Association (NMEA) sentences provided by any GPS device. These sentences contain the standard information required to get position, speed, and other parameters.

GPSID accesses these sentences through two interfaces:

- Parsed GPS Data interface—parses the NMEA sentences and associates them into a data structure called GPS_POSITION. This interface provides most of the standard GPS information, and also handles events when data arrives.
- Raw GPS Data interface—extracts NMEA sentences directly without parsing any data. This interface is useful in getting more information than through the parsed GPS data interface. For more information see Section 2.2, “Accessing Raw GPS Data.”

1 Settings

Basically, the GPSID works with two configurations: connecting GPSID to the GPS hardware and providing data to the applications that use the GPSID.

First add the GPS and GPSID Drivers to the catalog:

- Add to Catalog > Core OS > CEBASE > Application and Services Development > Location > GPS Intermediate Driver, as shown in Figure 2.

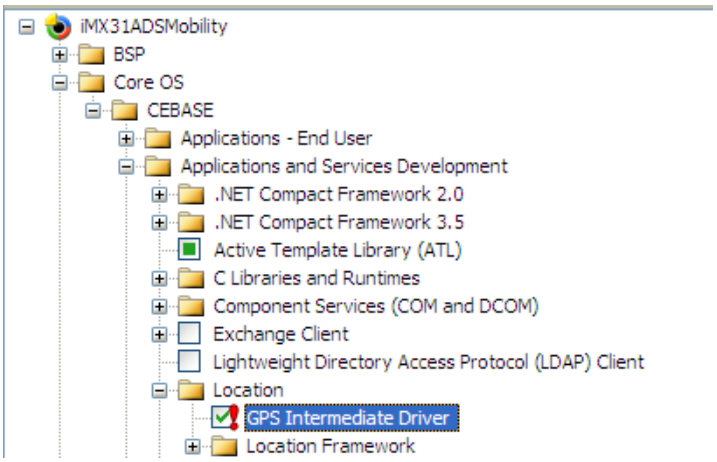

Figure 2. Adding GPS Intermediate Driver

- Add to Catalog > Third Party > BSP > Freescale i.MX31 3DS: ARMV4I > Device Drivers > GPS, as shown in Figure 3.

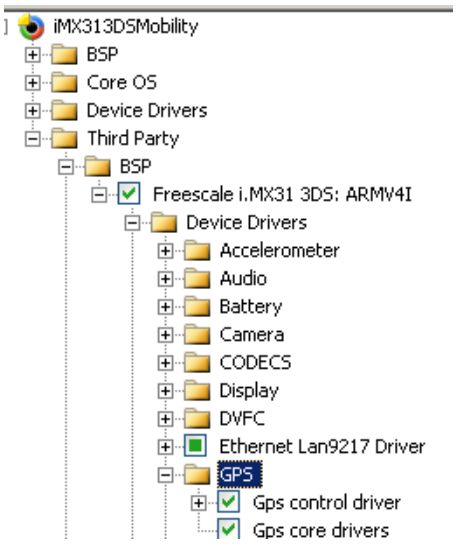

Figure 3. Adding GPS Drivers

1.1 Setting GPS Hardware Configuration

There are two main registry entries that must be set before the GPS hardware is connected.

- GPSID Input Source Registry Settings
- GPSID GPS Hardware Registry Settings

1.1.1 GPSID Input Source Registry Settings

This registry configures the input source from which it retrieves GPS data. Usually, this input source is a physical GPS hardware accessed by a COM port, Compact Flash Card, or some other input device.

The key is located at:

```
HKEY_LOCAL_MACHINE\System\CurrentControlSet\GPS Intermediate Driver\Drivers
```

The sub key for this entry is called CurrentDriver. This contains the connection information to be used for retrieving GPS location data. Each input source used by GPSID requires its own key beneath.

```
HKEY_LOCAL_MACHINE\System\CurrentControlSet\GPS Intermediate Driver\Drivers
```

For instance, if a GPS hardware is connected to a specific COM port, and this key entry is named as `MyGPSHardware`, the input source configuration settings are as follows:

```
HKEY_LOCAL_MACHINE\System\CurrentControlSet\GPS Intermediate Driver\Drivers\CurrentDriver
```

CurrentDriver—`MyGPSHardware`

```
HKEY_LOCAL_MACHINE\System\CurrentControlSet\GPS Intermediate
Driver\Drivers\CurrentDriver\MyGPSHardware
```

InterfaceType—COMM

FriendlyName—My GPS Hardware

In the first entry, the CurrentDriver value is defined as the name of the input source. In this case, the input source is named `MyGPSHardware`. In the second entry, the `MyGPSHardware` key is used. The InterfaceType of the GPS hardware, using the COM port interface, is also defined.

FriendlyName variable contains a human-readable name of the input source.

1.1.2 GPSID GPS Hardware Registry Settings

This registry configures the physical GPS hardware connection. After configuring the InterfaceType entry, the COM port is set. The settings, located under the input source have been defined previously and this affects only the respective input source. Based on the example, the settings must be under the following registry key:

```
HKEY_LOCAL_MACHINE\System\CurrentControlSet\GPS Intermediate Driver\Drivers\MyGPSHardware
```

The following list shows the registry entry that specifies GPS hardware connection information:

- CommPort—Name of the COM port to which the physical GPS hardware is connected.
- Baud—Baud rate at which the GPS hardware operates
- Parity—Parity scheme for COM port configuration
- StopBits—Stop bit for COM port configuration

There are entries that help in configuring additional features on the COM port. Please check the GPSID Windows help for more information.

1.2 Providing Data to Applications that use the GPSID

The GPSID can handle multiple client applications using internal multiplexing code that provides the same data to multiple clients. The DriverInterface registry entry defines the name of the device that applications use to connect when using ReadFile(). Applications that use the parse GPS data API retrieve the information using the GPSGetPosition(), irrespective of the interface they are connected.

NOTE

The DriverInterface registry has to be configured, only if the raw GPS data is accessed with different applications.

This registry entry must be configured under the key:

```
HKEY_LOCAL_MACHINE\System\CurrentControlSet\GPS Intermediate Driver\Multiplexer
```

DriverInterface—It is the name of the communication port that applications use to access GPS location information previously defined by the GPSID configuration. For example, the value might be COM2 or GPD2. This value must begin with COM and contain a number from 0 to 4294967295. Also, it can begin with GPD and contain a number from 1 to 4294967295.

MaxBufferSize—The GPSID stores the raw data from GPS hardware in a buffer. Be careful while defining the buffer size. If the application reads more slowly than the GPS hardware is writing into the buffer, the user may lose data.

Also, the user needs to add the following registry entry to the following key:

```
HKEY_LOCAL_MACHINE\System\CurrentControlSet\GPS Intermediate Driver\Multiplexer\ActiveDevice
```

This contains additional configuration settings for the GPSID multiplexer.

Flags—Device driver flags for the GPSID. These flags follow the conventions described for the device manager function ActivateDeviceEx().

2 Extracting GPS Information

There are two APIs for extracting information from a GPS device. The APIs are Parsed GPS Data and Raw GPS Data. The most frequently used is the parsed GPS data since it covers all basic sentence information and handles several connections to the same hardware.

2.1 Using Parsed GPS Data

This interface parses the NMEA sentences provided by the GPS, and associates them into a data structure called GPS_POSITION. The application accesses this data structure to get the information. The functions are described in the following sections:

2.1.1 GPSOpenDevice()

Creates a connection to the GPSID.

The function structure is as follows:

```
HANDLE GPSOpenDevice (HANDLE hNewLocationData, HANDLE hDeviceStateChange, const WCHAR
*szDeviceName, DWORD dwFlags).
```

The parameter of the function are as follows:

hNewLocationData	Handle to a Windows Embedded CE event created using CreateEvent. The GPSID signals the passed event whenever it has new GPS location information.
hDeviceStateChange	Handle to a Windows Embedded CE event created using CreateEvent. The GPSID signals the passed event whenever the state of the device changes.
szDeviceName	Reserved and must be NULL.
dwFlags	Reserved and must be 0.

This function returns to the GPSID if successful.

2.1.2 GPSGetPosition()

Retrieves GPS location information such as latitude, longitude, altitude, and speed. All these data are passed to the GPS_POSITION structure. The function structure is as follows:

```
DWORD GPSGetPosition (HANDLE hGPSDevice, GPS_POSITION *pGPSPosition, DWORD dwMaximumAge, DWORD
dwFlags).
```

The parameters of the function are as follows:

hGPSDevice	Handle returned by a call to GPSOpenDevice().
pGPSPosition	Pointer to a GPS_POSITION structure. This structure is filled with location data obtained by the GPSID.
dwMaximumAge	GPSID returns the received information within the time specified by this parameter.
dwFlags	Reserved and must be 0.

For more information on the GPS_POSITION data structure, refer the following Windows Help link:

http://msdn.microsoft.com/en-us/library/ms893674.aspx

2.1.3 GPSGetDeviceState()

This function retrieves information about the current state of the GPS hardware device. For more information on this function, refer the following Windows Help link:

http://msdn.microsoft.com/en-us/library/ms893612.aspx

2.1.4 GPSCloseDevice()

This function closes the connection to the GPSID. If there is only one connection to the GPSID, the driver turns off the GPS hardware to save power. The function structure is as follows:

```
DWORD GPSCloseDevice (HANDLE hGPSDevice)
```

hGPSDevice—Handle returned by a call to GPSOpenDevice() and returns ERROR_SUCCESS if successful.

Example 1 shows how to extract the latitude and longitude information from the parsed GPS data.

Example 1. Code to Extract Latitude and Longitude Information from GPS Data

```
#include "Gpsapi.h"
        HANDLE gpsHandler;
        double Latitude;
        double Longitude;

int _tmain(int argc, TCHAR *argv[], TCHAR *envp[]
{
        GPS_POSITION gpsPosition = NULL;
        gpsPosition.dwVersion = GPS_VERSION_1;
        gpsPosition.dwSize = sizeof(gpsPosition);

        gpsHandler = GPSOpenDevice(NULL, NULL, NULL, 0);

        if(GPSGetPosition (gpsHandler, &gpsPosition, 1000, 0) != ERROR_SUCCESS)
                return;

        Latitude = gpsPosition.dblLatitude;//Variable sets with Latitude
        Longitude = gpsPosition.dblLongitude;//Variable sets with Longitude
        GPSCloseDevice(gpsHandler);

        return 0;
}
```

2.2 Accessing Raw GPS Data

The raw GPS data interface accesses the NMEA sentences directly without parsing of data. This interface is used mostly in applications that read directly into the NMEA sentences. The application or some driver above performs parsing of all the NMEA sentences.

Execute the following steps to use the raw GPS data interface:

1. Call the CreateFile() to open a connection to the GPSID multiplexer. The first parameter on this function is the COMM port name connected to the GPS Hardware and defined on the registry entries.
2. Read the information from the GPSID calling ReadFile(). This information is a standard NMEA sentence.
3. Parse the NMEA sentences and extract the information the user is going to use.
4. Repeat steps 2 and 3 as needed to refresh information on your application.
5. Call the CloseHandle() to close the connection.

3 Revision History

Table 1 provides a revision history for this application note.

Table 1. Document Revision History

Rev. Number	Date	Substantive Change(s)
0	01/2010	Initial release.

Appendix A Standard NMEA Sentences

Some of the standard NMEA Sentences are discussed in the following sections.

A.1 $GPGGA - GPS Fix Data

Time, position, and fix-related data for a GPS receiver are shown in the following sentence:

```
$GPGGA,hhmmss.ss,llll.ll,a,yyyyy.yy,a,x,xx,x.x,x.x,M,x.x,M,x.x,xxxx*CC<CR><LF>
   1       2         3         4     5  6   7   8 9   A B   C   D
```

The values of each parameter are as follows:

- 1— Sentence identifier
- 2— Coordinated Universal Time (UTC) of position
- 3— Latitude of position (North/South)
- 4— Longitude of position (East/West)
- 5— GPS quality indicator
- 6— Number of satellites in view, 00–12
- 7— Horizontal dilution of precision
- 8— Antenna altitude above/below mean sea level (geoid)
- 9— Units of antenna altitude (meters)
- A— F.FFFUUU where: F.FFF is Reference frequency offset (parts per million) and UUU is Reference frequency offset uncertainty (parts per billion)
- B— Units of geoidal separation (meters)
- C— Age of differential GPS data
- D— Differential reference station ID, 0000–1023
- CC— checksum

A.2 $GPRMC - Recommended Minimum Specific GPS/TRANSIT Data

The recommended minimum specific GPS/Transit data are shown in the following sentence.

```
$GPRMC,154232,A,2758.612,N,08210.515,W,085.4,084.4,230394,003.1,W,*CC<CR><LF>
  1      2  3     4    5     6    7    8     9     A     B   C
```

The values of each parameter are as follows:

- 1— Sentence identifier
- 2— UTC time - 154232 denotes 15 hrs, 42 min, 32 sec. The seconds part may have a decimal.
- 3— Validity: A denotes good, V denotes not good
- 4— Latitude - 2758.612 denotes 27 degrees, 58.612 minutes
- 5— Hemisphere: N for NORTH, S for SOUTH
- 6— Longitude - 08210.515 denotes 082 degrees, 10.515 minutes
- 7— Hemisphere: W for WEST, E for EAST
- 8— Speed (knots)
- 9— Track - degrees true
- A— UTC date - 230394 denotes day 23, month 03, year 94
- B— Magnetic variation - degrees
- C— Variation direction - W for WEST (+), E for EAST (-)
- CC— checksum

A.3 $GPGSA - GSA - GPS DOP and Active Satellites

GPS DOP and active satellites are shown in the following sentence.

```
$GPGSA,A,3, 04,05,,09,12,,,24,,,,,  2.5, 1.3, 2.1, *39 <CR> <LF>
  1    2 3             4              5    6    7
```

The values of each parameter are as follows:

- 1— Sentence identifier
- 2— Selection of 2D or 3D fix, A denotes auto selection, M denotes manual selection
- 3— 3D fix, 1 denotes no fix, 2 denotes 2D fix
- 4— 04,05.... PRNs of satellites used for fix (space for 12)
- 5— 2.5 Position dilution of precision (PDOP)
- 6— 1.3 Horizontal dilution of precision (HDOP)
- 7— 2.1 Vertical dilution of precision (VDOP)
- CC— checksum

A.4 $GPGSV - GNSS Satellites in View

GPS Satellites in view are shown in the following sentence.

```
$GPGSV,x,x,xx,xx,xx,xxx,xx.............,xx,xx,xxx,xx,*CC <CR> <LF>
   1   2 3   4   5   6   7  8
```

The values of each parameter are as follows:

- 1— Sentence identifier
- 2— Message number, 1–9
- 3— Total number of messages, 1–9
- 4— Total number of satellites in view
- 5— Satellite ID number
- 6— Elevation in degrees, 90 degree is maximum
- 7— Azimuth in degrees, true if the range 000–359
- 8— SNR (C/No) 00–99 dB-Hz. Null when not tracking.
- CC— checksum

THIS PAGE INTENTIONALLY LEFT BLANK

How to Reach Us:

Home Page:
www.freescale.com

Web Support:
http://www.freescale.com/support

USA/Europe or Locations Not Listed:
Freescale Semiconductor, Inc.
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
1-800-521-6274 or
+1-480-768-2130
www.freescale.com/support

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
www.freescale.com/support

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku
Tokyo 153-0064
Japan
0120 191014 or
+81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor China Ltd.
Exchange Building 23F
No. 118 Jianguo Road
Chaoyang District
Beijing 100022
China
+86 10 5879 8000
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor
Literature Distribution Center
1-800 441-2447 or
+1-303-675-2140
Fax: +1-303-675-2150
LDCForFreescaleSemiconductor
@hibbertgroup.com

Document Number: AN4046
Rev. 0
01/2010