FirstGPS™ Starter Kit

User Guide

Version 1.0
Part Number 44326-00-ENG
Revision A
March 2001

Trimble™

Corporate Office

Trimble Navigation Limited
645 North Mary Avenue
Post Office Box 3642
Sunnyvale, CA 94088-3642
U.S.A.
Phone: +1-408-481-8940, 1-800-545-7762
www.trimble.com

Support Offices

For support in Europe, call:
+44.1256.746.239
or send a fax to:
+44.1256.760.148.

For support outside Europe, call:
+1-408-481-8786
or send a fax to:
+1-408-481-2011

Release Notice

This is the March 2001 release of the FirstGPS Starter Kit User Guide, part number 44326-00-ENG. It applies to version 1.0 of the FirstGPS API software.

The following limited warranties give you specific legal rights. You may have others, which vary from state/jurisdiction to state/jurisdiction.The following limited warranties give you specific legal rights. You may have others, which vary from state/jurisdiction to state/jurisdiction.

Hardware Limited Warranty

Trimble warrants that this Trimble hardware product (the “Product”) shall be free from defects in materials and workmanship and will substantially conform to Trimble’s applicable published specifications for the Product for a period of one (1) year, starting from the date of delivery. The warranty set forth in this paragraph shall not apply to software/firmware products.

Software and Firmware License, Limited Warranty

This Trimble software and/or firmware product (the “Software”) is licensed and not sold. Its use is governed by the provisions of the applicable End User License Agreement (“EULA”), if any, included with the Software. In the absence of a separate EULA included with the Software providing different limited warranty terms, exclusions and limitations, the following terms and conditions shall apply. Trimble warrants that this Trimble Software product will substantially conform to Trimble’s applicable published specifications for the Software for a period of ninety (90) days, starting from the date of delivery.

Warranty Remedies

Trimble's sole liability and your exclusive remedy under the warranties set forth above shall be, at Trimble’s option, to repair or replace any Product or Software that fails to conform to such warranty ("Nonconforming Product") or refund the purchase price paid by you for any such Nonconforming Product, upon your return of any Nonconforming Product to Trimble in accordance with Trimble’s standard return material authorization procedures.

Warranty Exclusions and Disclaimer

These warranties shall be applied only in the event and to the extent that: (i) the Products and Software are properly and correctly installed, configured, interfaced, maintained, stored, and operated in accordance with Trimble's relevant operator's manual and specifications, and; (ii) the Products and Software are not modified or misused. The preceding warranties shall not apply to, and Trimble shall not be responsible for defects or performance problems resulting from (i) the combination or utilization of the Product or Software with products, information, data, systems or devices not made, supplied or specified by Trimble; (ii) the operation of the Product or Software under any specification other than, or in addition to, Trimble's standard specifications for its products; (iii) the unauthorized modification or use of the Product or Software; (iv) damage caused by accident, lightning or other electrical discharge, fresh or salt water immersion or spray; or (v) normal wear and tear on consumable parts (e.g., batteries).

THE WARRANTIES ABOVE STATE TRIMBLE'S ENTIRE LIABILITY, AND YOUR EXCLUSIVE REMEDIES, RELATING TO PERFORMANCE OF THE PRODUCTS AND SOFTWARE. EXCEPT AS OTHERWISE EXPRESSLY PROVIDED HEREIN, THE PRODUCTS, SOFTWARE, AND ACCOMPANYING DOCUMENTATION AND MATERIALS ARE PROVIDED "AS-IS" AND WITHOUT EXPRESS OR IMPLIED WARRANTY OF ANY KIND BY EITHER TRIMBLE NAVIGATION LIMITED OR ANYONE WHO HAS BEEN INVOLVED IN ITS CREATION, PRODUCTION, INSTALLATION, OR DISTRIBUTION INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, TITLE, AND NONINFRINGEMENT. THE STATED EXPRESS WARRANTIES ARE IN LIEU OF ALL OBLIGATIONS OR LIABILITIES ON THE PART OF TRIMBLE ARISING OUT OF, OR IN CONNECTION WITH, ANY PRODUCTS OR SOFTWARE. SOME STATES AND JURISDICTIONS DO NOT ALLOW LIMITATIONS ON DURATION OR THE EXCLUSION OF AN IMPLIED WARRANTY, SO THE ABOVE LIMITATION MAY NOT APPLY TO YOU.

TRIMBLE NAVIGATION LIMITED IS NOT RESPONSIBLE FOR THE OPERATION OR FAILURE OF OPERATION OF GPS SATELLITES OR THE AVAILABILITY OF GPS SATELLITE SIGNALS.

Limitation of Liability

TRIMBLE'S ENTIRE LIABILITY UNDER ANY PROVISION HEREIN SHALL BE LIMITED TO THE GREATER OF THE AMOUNT PAID BY YOU FOR THE PRODUCT OR SOFTWARE LICENSE OR U.S.$25.00. TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, IN NO EVENT SHALL TRIMBLE OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES WHATSOEVER UNDER ANY CIRCUMSTANCE OR LEGAL THEORY RELATING IN ANY WAY TO THE PRODUCTS, SOFTWARE AND ACCOMPANYING DOCUMENTATION AND MATERIALS, (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION, LOSS OF BUSINESS INFORMATION, OR ANY OTHER PECUNIARY LOSS), REGARDLESS WHETHER TRIMBLE HAS BEEN ADVISED OF THE POSSIBILITY OF ANY SUCH LOSS AND REGARDLESS OF THE COURSE OF DEALING WHICH DEVELOPS OR HAS DEVELOPED BETWEEN YOU AND TRIMBLE. BECAUSE SOME STATES AND JURISDICTIONS DO NOT ALLOW THE EXCLUSION OR LIMITATION OF LIABILITY FOR CONSEQUENTIAL OR INCIDENTAL DAMAGES, THE ABOVE LIMITATION MAY NOT APPLY TO YOU.

Contents

3 Software Integration

A Using the FirstGPS API

About This Manual

Welcome to the *FirstGPS Starter Kit User Guide*. This manual describes how to install, set up, and use the FirstGPS™ Application Programming Interface (API).

If you are not familiar with the Global Positioning System (GPS), visit our Web site for an interactive look at Trimble and GPS at:

- www.trimble.com

Trimble assumes that you are familiar with Microsoft Windows and know how to use a mouse, select options from menus and dialogs, make selections from lists, and refer to online help.

Related Information

This manual is provided in portable document format (PDF).

Other sources of related information are:

- Help – The software has built-in, context-sensitive help that lets you quickly find the information you need. Access it from the *Help* menu. Alternatively, click the **Help** button in a dialog, or press [F1]. To access the What's This? Help, click the question mark in the top right corner of a dialog box then click the relevant item.
- Readme.txt file – A Readme.txt file contains information added after the documentation was completed. To read this file, double-click it or use a text editor to open it. The installation program also copies this file into the program directory.
- Release notes – The release notes describe new features of the product, information not included in the manual, and any changes to the manual. The release notes are provided as a .doc file on the CD and are installed in the program directory (typically C:\Program Files\Trimble\FirstGPS) when you install the software. Use a text editor to view the contents of the release notes.
- ftp.trimble.com – Use the Trimble FTP site to send files or to receive files such as software patches, utilities, service bulletins, and FAQs. Alternatively, access the FTP site from the Trimble Web site at www.trimble.com/support/support.htm.
- Trimble training courses – Consider a training course to help you use your GPS system to its fullest potential. For more information, visit the Trimble Web site at:
  - www.trimble.com/support/training.htm

Technical Assistance

If you have a problem and cannot find the information you need in the product documentation, do one of the following:

- Send an e-mail to:

  firstgps_support@trimble.com

- Contact support at one of the numbers listed in the front section of this manual.
- Request technical support using the Trimble Web site at:

  www.trimble.com/support/support.htm

Your Comments

Your feedback about the supporting documentation helps us to improve it with each revision. To forward your comments, do one of the following:

- Send an e-mail to:

  ReaderFeedback@trimble.com.

- Complete the Reader Comment Form at the back of this manual and mail it according to the instructions at the bottom of the form.

If the reader comment form is not available, send comments and suggestions to the address in the front of this manual. Please mark it *Attention: Documentation Group*.

Document Conventions

The document conventions are as follows:

Convention	Definition
Italics	Identifies software menus, menu commands, dialog boxes, and the dialog box fields.
Helvetica Narrow	Represents messages printed on the screen.
Helvetica Bold	Identifies a software command button, or represents information that you must type in a software screen or window.
'Select *Italics* / *Italics*'	Identifies the sequence of menus, commands, or dialog boxes that you must choose in order to reach a given screen.
[Ctrl]	Is an example of a hardware function key that you must press on a personal computer (PC). If you must press more than one of these at the same time, this is represented by a plus sign, for example, [Ctrl]+[C].

CHAPTER

1

FirstGPS Starter Kit Overview

In this chapter:

- Purpose of the Starter Kit
- FirstGPS Architecture
- Starter Kit Components

Purpose of the Starter Kit

FirstGPS™ technology is designed for customers who need a simple Global Positioning System (GPS) chipset in their applications.

The Starter Kit is for evaluation of FirstGPS software performance. The Starter Kit includes:

- A sample hardware board
- A sample hardware environment, including antenna, power, and serial communications
- A sample Windows software application (Microsoft® Windows 95 / Windows 98 / Windows NT / Windows 2000)
- Sample source code for a typical application with examples of API calls
- This manual in PDF format on the supplied CD

Together, these give a preliminary view of the hardware and software interfaces and provide the basis for a high-level discussion of hardware and software integration issues.

FirstGPS Architecture

With FirstGPS software, the Position-Velocity-Time (PVT) solution is accessed directly by the host application through calls to an Application Programming Interface (API). This is a significant departure from past practice, where the GPS function was usually a ***PVT-serial solution***, that is, a complete GPS receiver board that generated the Position Velocity Time (PVT) fix, and then communicated the PVT fix to the host application through a serial port. The API-based design offers considerably more flexibility for the applications designer, along with benefits in terms of size, power, and cost.

The FirstGPS architecture is built around three components designed by Trimble:

- Colossus™ Radio Frequency (RF) ASIC
- IO Digital Signal Processing (DSP) ASIC
- FirstGPS software library[1]

The Colossus and IO[2] chips are the center of the GPS function hardware core. The FirstGPS library is the software interface with the GPS function hardware core, and it is linked into the customer's software application.

A complete FirstGPS implementation comprises the following components:

- Power
- Antenna and low-noise amplifier (LNA)
- The GPS function hardware core
- The host software application (Application Layer and FirstGPS library)

---

1. The FirstGPS software library is sometimes referred to as simply the FirstGPS library.

2. The IO chip (pronounced EEH-ooh) is named after the moon of Jupiter, whose discovery (along with the three other Galilean moons) was one of the landmarks in the history of navigation. By marking the Jovian lunar transits/eclipses (Galileo estimated there are three per day on average), two navigators can easily synchronize their timepieces. This established the first *anywhere*, *anytime* time standard. Since the measurement of time is equivalent to the measurement of longitude, observation of the Jovian moons allowed absolute accurate positioning for the first time in the history of man. For a more detailed account of Galileo's discovery, and the invention of the chronometer which *eclipsed* his discovery, see ***Longitude: The True Story of a Lone Genius Who Solved the Greatest Scientific Problem of His Time,*** Dava Sobel, 1995.

Input/Output Interface

The Starter Kit contains a complete sample FirstGPS implementation, described in Starter Kit Components, page 9. The sample hardware core is the Measurement Platform Module (MPM), described on page 9. The sample host software application is a Windows-based program described in detail in Chapter 2, Using the Starter Kit. The sample power interface, antenna interface, and the serial interface between MPM and host CPU are provided by the motherboard housing the MPM.

FirstGPS Hardware Core

There are two FirstGPS ASICs, an RFIC (Colossus) and a DSP ASIC (IO). These two chips—combined with a voltage regulator, a Surface Acoustic Wave (SAW) filter, and a crystal oscillator—create the GPS function hardware core. Figure 1.1 shows the block diagram of this hardware core.

The MPM in the FirstGPS Starter Kit is a sample board layout of the GPS hardware component. The MPM board is described in Measurement Platform Module (MPM), page 9. Layout depends upon the customer's application. Proper board layout is crucial to good RF performance.

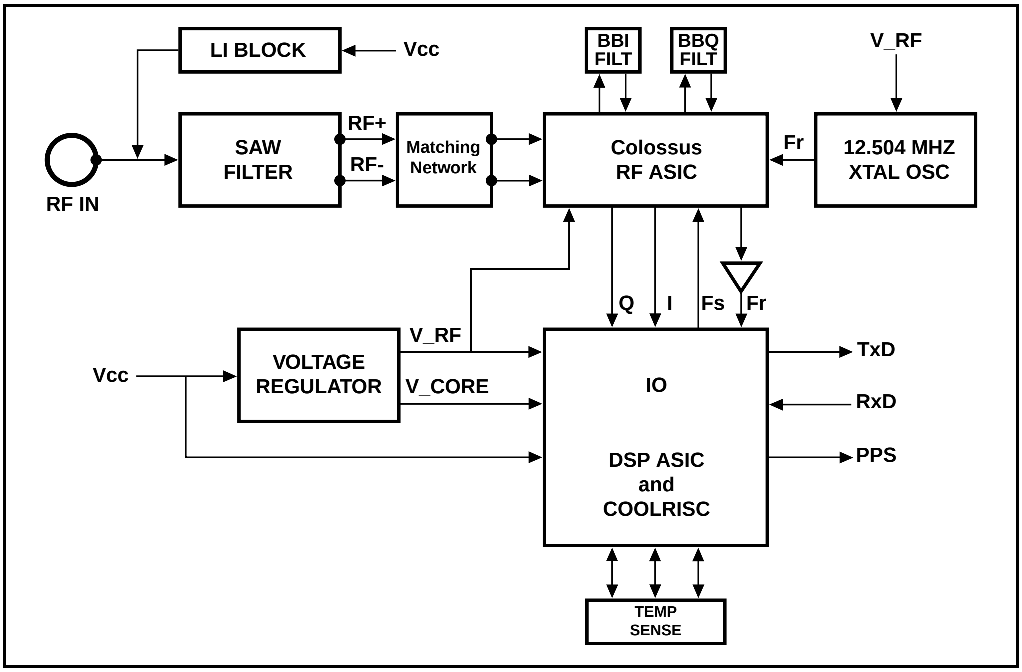


Figure 1.1 Colossus / IO Block Diagram

Power

The hardware core requires three voltages:

Hardware Core	Voltage
VCC	3.3 ±0.3 VDC
V_RF	3.0 ±0.3 VDC
V_CORE	2.0 ±0.2 VDC

In a typical implementation, 3.3 VDC is supplied to VCC and a voltage regulator distributes the voltage to the various components as required.

Oscillator

The Local Oscillator (LO) for FirstGPS can be either 12.504 MHz or 13.000 MHz. The crystal must be accurate to within 10 PPM. Trimble recommends a Temperature-Compensated Crystal Oscillator (TCXO).

Colossus RFIC

The GPS signal conversion from an analog signal to digital data is performed by the Colossus RF ASIC. This ASIC contains a two-stage down-converter using a 1400 MHz VCO located inside the chip. The first conversion brings the GPS signal from 1575.42 MHz (L1) to an Intermediate Frequency (IF) of 175.42 MHz. The second conversion is to the pseudo-base band frequency of 84 kHz. After the second conversion, the GPS signal is divided into ***I*** and ***Q*** bits which are in phase-quadrature with respect to each other. The ***I*** and ***Q*** bits are passed through a low-pass filter off chip, brought back in to a limiting amplifier, and then sampled before being passed onto the IO DSP. The sampling frequency input to the Colossus, by the IO DSP, is the external LO frequency divided by four. For a 12.504 MHz system the sampling frequency would be 3.126 MHz.

Note – *Keep all of the above frequencies in mind for EMI purposes when doing system design.*

All three voltages listed in Power, page 5 are supplied to Colossus to feed the separate sections of the IC, VCC_RF for the RF conversion, VCC_DIG for the digital dividers, and VCC_OUT for the ***I*** and ***Q*** output buffers.

IO DSP

The GPS signal-processing function is performed by the IO DSP. This ASIC contains the GPS correlation channels and a small CoolRISC processor core with its own RAM and ROM. The CoolRISC runs the tracking loop code that controls the correlators.

The FirstGPS architecture is different from most GPS receiver architectures. In all modern GPS architectures, the software resides in a multitasking environment. The tasks perform a variety of functions, including:

- tracking loop control
- measurement conditioning
- position computations
- navigation data decoding

The highest priority task is the signal-processing code (tracking loops). This code, although it has a small code footprint, has a very high interrupt rate (up to 2000 Hz) with a stringent latency requirement. The rest of the tasks can loosely be grouped as ***navigation code***. These form a larger body of code, heavy with floating-position operations, but with higher tolerance to latency.

In typical GPS board implementations, all tasks are hosted in the same CPU even though they have significantly different characteristics. In contrast, the FirstGPS architecture hosts the signal-processing function on the IO and the navigation function on a host CPU, allowing the processor characteristics to be optimized for each function and reducing total system loading.

IO Interfaces

The outputs of the IO tracking loops are sent via serial link to the host CPU, where the FirstGPS navigation code resides. There, the raw outputs from IO are processed into range, Doppler, and orbit data information, and then further processed into a PVT solution. These results are made available through a fully featured FirstGPS API to the host application. In the reverse direction, IO receives tracking loop controls from the FirstGPS software via the serial link. The serial link rate between IO and host CPU is typically 19200 baud.

The IO has a Pulse-per-Second (PPS) function. One of the IO pins is held high for approximately 5 microseconds at the start of every UTC second. An API call is available to apply a small time bias, typically an advance of a few microseconds, to account for cable lengths.

The IO also has a temperature function. Inside the IO chip is a temperature-sensitive counter. An API call provides the counts in real-time, and the counts can be converted into temperature using a simple formula.

FirstGPS Software

The FirstGPS software library is the GPS function software core. It resides on the host CPU, so it shares RAM and ROM with the customer's application. The library must be compiled specific to each application, depending on the CPU, Real-time Operating System (RTOS), and compiler. The source code is very portable, written in C, and it has been ported to a variety of different development environments. The FirstGPS software library encompasses the complete software interface with the IO chip, so the IO is essentially transparent to the application software.

A complete list of API calls for the FirstGPS software library is listed in Appendix A, Using the FirstGPS API. In general, the application program uses the FirstGPS API calls to:

- Boot the IO and start the GPS function.
- Initialize the receiver with non-volatile memory and current time, if available.
- Change receiver control parameters (satellite masks, DGPS corrections).
- Fetch current receiver status.
- Fetch the most recent PVT solution.
- Fetch ancillary data (for example, signal strengths).
- Stop the GPS function.

Application Layer

The term ***Application Layer*** is used to describe the host application software. This is typically software running a navigation system, a cellular wireless device, or any of a number of devices that benefit from knowledge of position, running on a Real-time Operating System (RTOS). This software provides at least the boot code and the user interface, and it exercises the FirstGPS software library through the function calls contained in the FirstGPS API.

Starter Kit Components

Measurement Platform Module (MPM)

The MPM board is a Trimble assembly designed for optimal performance of the IO DSP and Colossus RF ASICs running at a reference frequency of 12.504 MHz. A dual-output voltage regulator is used to create the VCC_RF and VCC_Core needed to run both the Colossus and IO DSP chips. The board size is a standard Trimble form-factor of 32 mm x 66 mm. The antenna connector is a right-angle, board-mounted MCX connector. I/O interfacing is through an 8-pin male header with the following pin identities:

Pin #	Function	Pin #	Function
1	No Connect	5	RXD
2	Power	6	PPS
3	TXD	7	SP_EN (Serial Port Enable)
4	VCC_Ant (power for 5 V antenna)	8	GND

Figure 1.2 shows the top layout of Trimble's MPM assembly:

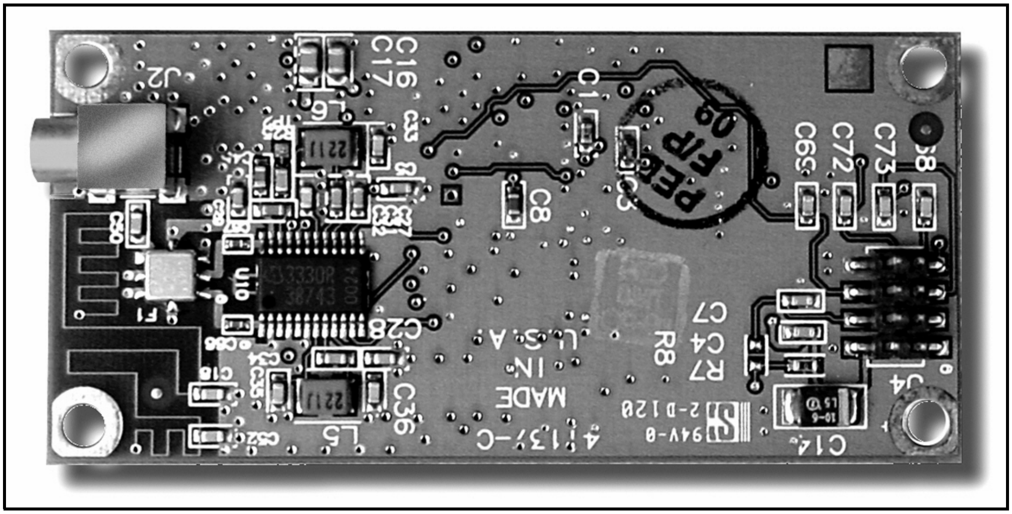


Figure 1.2 Top MPM Assembly Layout

Figure 1.3 shows the bottom layout of Trimble's MPM assembly:

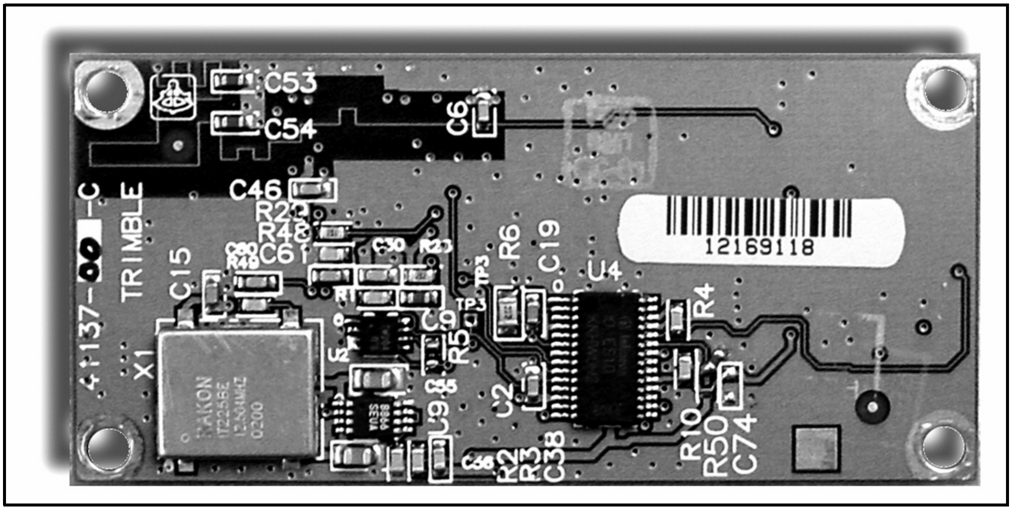


Figure 1.3 Bottom MPM Assembly Layout

Motherboard

The MPM interface motherboard includes a +9 to +32 VDC input software switching power supply that provides a regulated +3.3 VDC to the MPM receiver. It also converts the TTL level I/O to RS-232 for a direct interface to the computer. The motherboard provides an open collector interface for the PPS. The input voltage must be between +9 VDC and +32 VDC and is input to the board through a 3-pin circular connector of which only two pins are used. Supplying power outside the specified input range will damage the board.

Although there are two communication ports on the motherboard, only one is available for use by the MPM (Port 1).

The pin identities for PORT 1 are:

Pin #	Function
1	NC
2	TX
3	RX
4	NC
5	GND
6	NC
7	NC
8	NC
9	PPS

Note – *Due to the open-collector interface, through the 9-pin RS-232 port, the polarity of the PPS signal is inverted. The pulse is a 4 µs negative-going pulse with the falling edge synchronized to UTC. When removed from the motherboard, the receiver provides a positive-going TTL level pulse, with the rising edge synchronized to UTC.*

Antenna

The Antenna supplied with the Starter Kit is a 3.3 VDC patch antenna with integrated cable and connector for terminating to the interface unit. This is an active antenna with +28 dB of gain. The MPM requires approximately 10 dB to 14 dB of gain at the antenna connector on board for best performance.

Power Supply

There are two ways to power the MPM Starter Kit using either the DC power cable or the AC/DC converter.

For DC:

1. Connect the terminated end of the power cable to the power connector on the interface unit.
2. Connect the red lead to positive voltage (+9 to +32 VDC).
3. Connect the black lead to GND.

   The yellow lead is not used.

For AC:

1. Connect the output cable of the converter to the 3-pin power connector on the interface unit.
2. Using an appropriate three-prong AC power cable (not provided), connect the converter to an AC wall socket (110 VAC or 220 VAC).

Software

See Chapter 2, Using the Starter Kit.

CHAPTER

2

Using the Starter Kit

This chapter includes the following topics:

- Touring the Sample Application
- Running the FirstGPS Starter Kit for the First Time
- Running the FirstGPS Starter Kit
- FirstGPS Monitor Main User Interface
- FirstGPS Monitor Features
- Configuring FirstGPS Monitor for the Output Protocols

Touring the Sample Application

This section provides a brief overview of the FirstGPS Starter Kit and Monitor program.

The Monitor program is a graphical user interface (GUI) application, which exercises the Trimble FirstGPS software library to generate position, time, and velocity information in a user-friendly visual Windows environment. The program allows performance evaluation of the FirstGPS platform running on the Microsoft Windows 95, 98, NT or 2000 operating environment.

Running the FirstGPS Starter Kit for the First Time

This section describes how to set up the FirstGPS Starter Kit. (The FirstGPS Monitor program is referred to as ***the GUI***.)

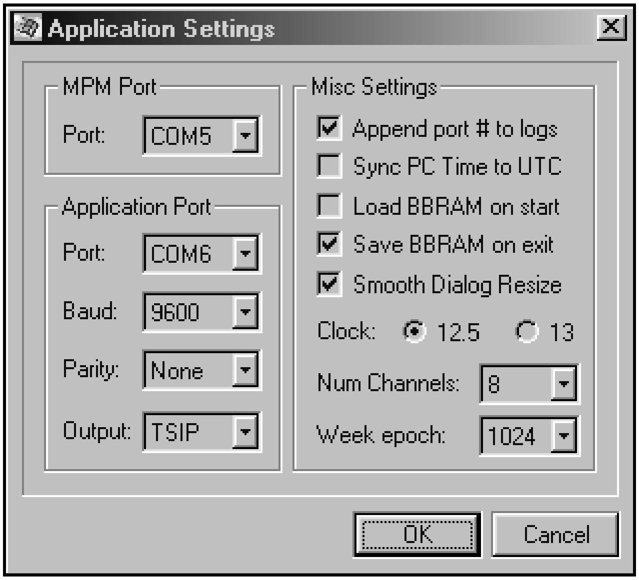
To set up the FirstGPS Starter Kit:

1. Connect the motherboard to a 9–32 VDC power supply.
2. Attach the antenna cable to the connector on the motherboard and place the antenna where over half the sky is visible.

   Note – *Windows with reflective glass or wire mesh can sometimes block the GPS signal.*
3. Use an RS-232 cable to connect Port 1 on the motherboard to a serial COM port on your PC.
4. Run the GUI.

   The FirstGPS Starter Kit CD installs the GUI to your C:\Program Files\Trimble\FirstGPS folder.

If this is the first time the program has been run, the application settings window shown below appears.



Be sure to select the correct application settings before running the MPM kit:

1. Set the *MPM Port* to the name of the COM port on the PC used to connect with the MPM.
2. Select the *Save BBRAM on exit* check box.

   This will provide a faster Time to First Fix (TTFF).
3. Set *Clock* to 12.504 MHz.
4. Set *Num Channels* to 12.
5. Set *Week epoch* to 1024.

   This includes all dates between August 22, 1999 and March 2019.

To modify these or other application settings at a later time, from the *FirstGPS Monitor* menu, page 17, select *Configure / Settings*.

Running the FirstGPS Starter Kit

When the *FirstGPS Monitor* main window appears, shown on page 17, the GPS engine is ready to start.

To launch the GPS engine (FirstGPS library):

- Select *Initialize / Start GPS*.

  If the application fails to initialize, a startup failed message is displayed in the status bar on the bottom left of the main window. If this occurs, please check that:

  – The COM port settings are correct in the application settings.

  – The serial cable is connected between Port 1 of the MPM and the PC's COM port.

  – The MPM kit is powered up with a correct DC power source.

To stop the GPS function after the GUI has been running:

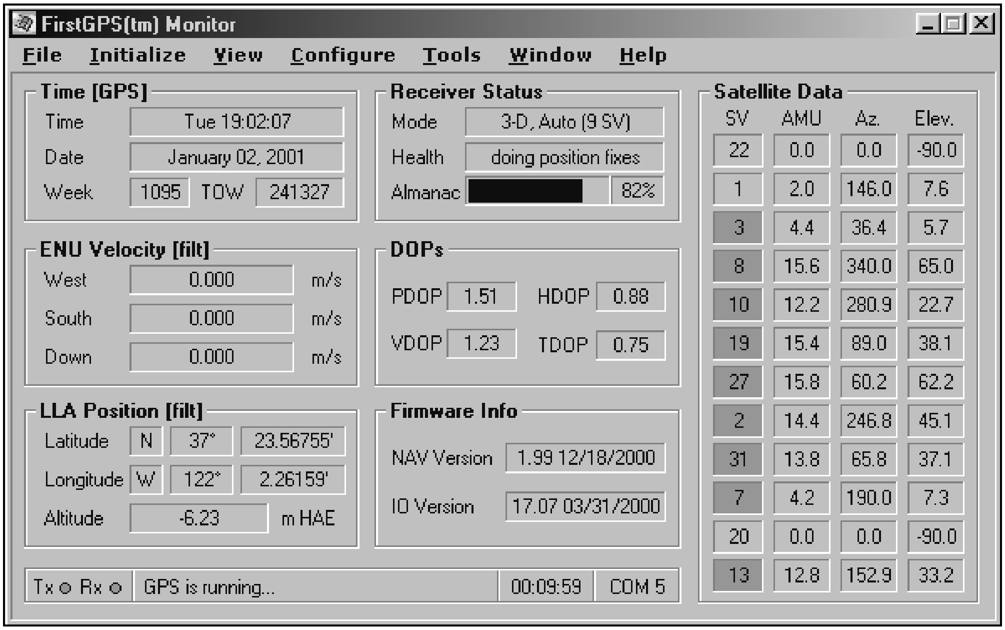
- Select *Initialize / Stop GPS*.

  This shuts down the FirstGPS library and the MPM. You can then exit the GUI.

Note – *If the GUI is still running when the MPM unit is disconnected from the power source, you must restart the GUI application.*

FirstGPS Monitor Main User Interface

The *FirstGPS Monitor* main window of the program, shown below, is divided into eight sections. Once the GPS is running and the program is communicating with the MPM kit, these sections are updated dynamically with current GPS information.



Time [GPS]

The *Time* panel displays the following, relative to GPS or UTC depending on the output options. (Refer to ***Timing Output***, page 23.)

- *Time* – the day of the week, and time in hours: minutes: seconds
- *Date* – month, day, and year
- *Week* and *TOW* – the current GPS week number and **t**ime **o**f **w**eek respectively.

For more information, refer to ***navGetCompactTime***, page 61.

ENU Velocity [filt]

ENU Velocity is the current velocity (updated every second) in the East-North-Up directions, in meters per second. For more information, refer to ***navGetLastFix***, page 66.

LLA Position [filt]

LLA Position is the current position (updated every second) in Latitude-Longitude-Altitude coordinates. Units are degrees and minutes relative to the WGS-84 datum. For latitude, *N* and *S* denote North and South; for longitude, *E* and *W* denote East and West. Altitude is computed in meters, referenced either to mean sea level or to the WGS-84 ellipsoid depending on the output options. For more information, refer to ***navGetLastFix***, page 66.

Receiver Status

Receiver Status displays current receiver information:

- *Mode* – the operating mode: 2D or 3D, Automatic or Manual
- *Health* – the health of the FirstGPS library
- *Almanac* – the status of the collected GPS almanac pages

The window also shows how much of the almanac has been collected and displays the message complete & current when the almanac collection has been completed. For more information, refer to ***navGetNavStatus***, page 68, and ***navGetSvSelections***, page 75.

DOPs

DOPs is the abbreviation for **D**ilution **O**f **P**recision numbers. These numbers are roughly associated with accuracy: the larger the numbers are, the less accurate the fix is.

The two most useful numbers are Horizontal DOP (HDOP), proportional to horizontal position accuracy, and Vertical DOP (VDOP). Position DOP (PDOP) is for three-dimensional position accuracy, and Time DOP (TDOP) is used for time accuracy. For more information, refer to ***navGetLastFix***, page 66, and to the Glossary.

Firmware Info

Firmware Info is the firmware version of the FirstGPS library and the firmware version of the MPM. For more information, refer to ***navGetNavVersion***, page 69.

Satellite Data

Satellite Data is the current satellite tracking information for each IO channel:

- *SV* – satellite identifier (PRN). If the PRN is against a green background, that satellite is usable for fixes.
- *AMU* – the signal strength (in AMUs)
- *Az.* – azimuth
- *Elev.* – elevation in degrees, if known. When elevation is unknown, the value `90` or `–90` is displayed.

For more information, refer to ***navGetChStatus***, page 60 and ***navGetSvSelections***, page 75.

Program Status

This section of the window has no title. It is located at the bottom of the window. Program Status indicates:

- Status of the serial connection between PC and MPM
- Status of the FirstGPS library
- Time since power-up
- Name of the COM port on the PC

FirstGPS Monitor Features

The menu bar in the FirstGPS Monitor main window, page 17, contains seven menus, as shown below.

The following sections describe these menus.

File Menu

Use *File / Exit* to close the *FirstGPS Monitor* window.

Initialize Menu

The following sections describe the six *Initialize* menu items.

Start / Stop GPS

When the GPS engine is not running, this menu item displays *Start GPS*. To start the GPS engine, select this item.

When the GPS engine is running, this menu item displays *Stop GPS*. To stop the GPS engine, select this item. *Stop GPS* will also save important parameters to Battery Backed-up RAM (BBRAM).

Cold Reset

Select *Initialize / Cold Reset* to restart the GPS engine and perform a firmware reset after erasing all of the collected GPS data such as the almanac and ephemeris. Cold resets result in a TTFF of 2 to 3 minutes.

Warm Reset

Select *Initialize / Warm Reset* to restart the GPS engine and perform a firmware reset after erasing the ephemeris from the collected GPS data. Performing a warm reset assumes that you have operated the MPM long enough to collect almanac information and calculate PVT. A typical warm reset results in a TTFF of about 50 seconds.

Hot Reset

Select *Initialize / Hot Reset* to restart the GPS engine and perform a firmware reset without erasing any collected GPS data. Performing a hot reset assumes that you have operated the MPM long enough to collect almanac and ephemeris information and calculate PVT. A typical hot reset results in a TTFF of about 15 seconds.

Set 2D Ref Altitude

Select *Initialize / 2D Ref Altitude* to prompt for a reference altitude for the receiver while operating in the 2D mode.

PPS Output

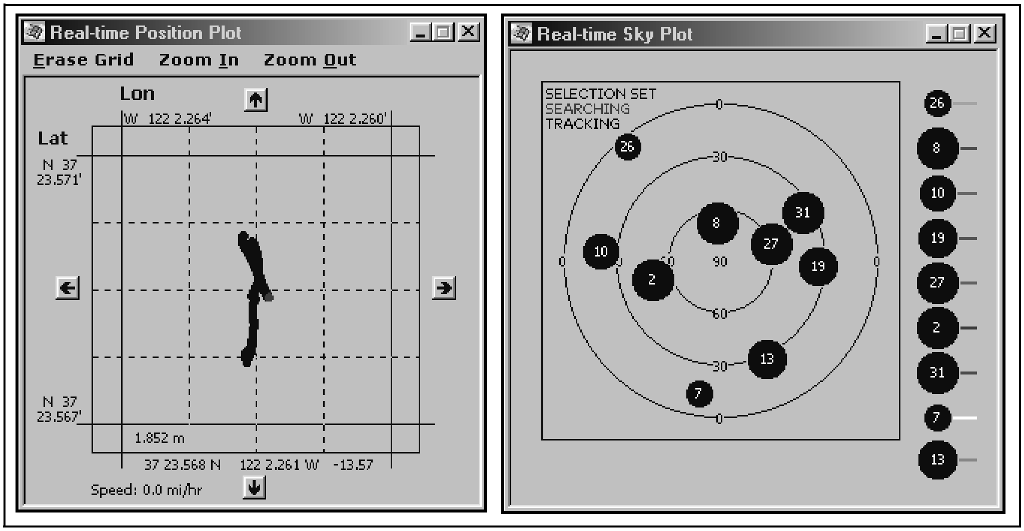
Select *Initialize / PPS Output* to enable the PPS output on the MPM board.

View Menu

The following sections describe the sixteen *View* menu items.

Plots

Select *View / Plots* to display a *Real-time Position Plot* and a *Real-time Sky Plot*.



Protocol Output

Select *View / Protocol Output* to view output data in the industry-standard NMEA protocol in a separate window. Click **Configure** in the output window to set up the protocol output options.

When you select to view NMEA output, the window displays ASCII NMEA sentences.

Position Output

Select *View / Position Output* to change the output position format, either latitude / longitude / altitude or Cartesian Earth-Centered-Earth-Fixed (ECEF) coordinates.

Velocity Output

Select *View / Velocity Output* to change the output velocity format, either East / North / Up or ECEF coordinates.

Timing Output

Select *View / Timing Output* to change the output time format, either GPS or UTC reference. The difference between UTC and GPS is equal to the number of leap seconds added since 1970.

Signal Level Output

Select *View / Signal Level Output* to change the satellite signal level output format to either AMU or CNO. For definitions of these terms, see page 83.

The signal level is displayed in the *Satellite Data* section of the main application window.

Almanac Pages

Select *View / Almanac Pages* to display GPS data for each of the collected satellite almanac pages. Refer to the *ICD-GPS-200* specification for definitions of the almanac terms.

Almanac Health

Select *View / Almanac Health* to display GPS data for the almanac health page. For more information, refer to the ***npNavStatusType***, ***ulAlmHealthStatus*** parameter on page 47.

Ephemeris Pages

Select *View / Ephemeris Pages* to display GPS data for each of the collected satellite ephemeris pages. Refer to the *ICD-GPS-200* specification for definitions of the ephemeris terms.

Ionosphere Info

Select *View / Ionosphere Info* to display GPS data for the ionospheric model data page. Refer to the *ICD-GPS-200* specification for definitions of the ionosphere information terms.

UTC Page Info

Select *View / UTC Page Info* to display GPS data for the UTC data page. Refer to the *ICD-GPS-200* specification for definitions of the UTC data terms.

GPS System Message

Select *View / GPS System Message* to display GPS system messages for each satellite. Refer to the *ICD-GPS-200* specification for definitions of the GPS system messages.

Oscillator Offset

Select *View / Oscillator Offset* to display the MPM’s oscillator bias and drift (updated every second). For more information, refer to ***navGetOscStatus***, page 70.

Channel Tracking Status

Select *View / Channel Tracking Status* to hide or display the *Satellite Data* section in the main window of the program. For more information, refer to ***navGetChStatus***, page 60 and ***navGetSvSelections***, page 75.

Satellite Visibility

Select *View / Satellite Visibility* to display the *Satellite Visibility* window which shows elevation angles of all satellites.

Temperature

Select *View / Temperature* to start the IO's temperature sensor reports. For more information, refer to ***navStartTempMeas***, page 78 and ***navGetTempParams***, page 76.

Configure Menu

The following sections describe the five *Configure* menu items.

Receiver Configuration

Select *Configure / Receiver Configuration* to configure the FirstGPS firmware parameters. For more information, refer to ***navGetNavConfig***, page 67.

Filter Configuration

Select *Configure / Filter Configuration* to configure the FirstGPS filter parameters.

Protocol Configuration

Select *Configure / Protocol Configuration* to configure NMEA protocol output options. For information on how to enable NMEA output, see Configuring FirstGPS Monitor for the Output Protocols, page 29.

Data Logging

Select *Configure / Data Logging* to configure a data logging option. You can log position fix data to a file in two different formats:

- Tab-delimited ASCII text
- Industry-standard NMEA ASCII text

Settings

Select *Configure / Settings* to configure the FirstGPS Monitor parameters. The descriptions of the parameters in the *Application Settings* window follow:

- *MPM Port* – the serial communication port number to which the MPM unit is connected through the RS-232 interface.
- *Application Port* – The output protocols port settings. This must be a port different from the MPM port. The PC must have at least one more serial port (in addition to the one used for the MPM port) to use the Output Protocols feature.
- *Append port # to logs* – Select this check box to append the MPM port number to the log file names, if any log options are enabled through the *Configure | Data Logging* menu item.
- *Sync PC time to UTC* – Select this check box to synchronize the local PC time to UTC, once the MPM unit has started running and the exact time is known.
- *Load BBRAM on start* – This option enables loading the BBRAM file to the FirstGPS library when connecting to the MPM unit. The BBRAM file contains the GPS system data such as:
  - the almanac
  - ephemeris
  - UTC
  - IONO(sphere) information

  Loading the current BBRAM file allows for faster TTFF.

This file does not come with the distribution of the FirstGPS Monitor application. Rather, it is generated automatically when the application runs for the first time and the *Save BBRAM on exit* option is selected. Usually, it takes about 15 minutes of tracking GPS signals before the BBRAM file is completely filled with GPS almanac pages. On consecutive starts, if the FirstGPS Monitor has been properly stopped (using the *Initialize / Stop* menu item), the BBRAM file is saved in the same directory as the GUI executable. Having this file available at the next power-up makes the TTFF much faster.

- *Save BBRAM on exit* – This option enables the generation of the BBRAM file as described in the previous paragraph.
- *Smooth Dialog Resize* – This option enables a smooth slide-out effect when various GUI dialog boxes are displayed.
- *Clock* – This option enables selection of the LO to 12.504 MHz or 13.000 MHz. The MPM in the Starter Kit comes with a 12.504 MHz crystal. The FirstGPS library will not run unless the *Clock* option is properly set.
- *Num Channels* – This option enables selection of the number of channels of the MPM unit. The MPM comes with 12 channels.
- *Week Epoch* – This option enables selection of the week epoch number, which is the number of 1024 week periods since 6 January 1980.

Tools Menu

The following sections describe the three *Tools* menu items.

TTFF Test Setup

Select *Tools / TTFF Test Setup* to configure and run a time-to-first-fix (TTFF) test.

TTFF Statistics

Select *Tools / TTFF Statistics* to display TTFF statistics.

Position Accuracy

Select *Tools / Position Accuracy* to display the position accuracy relative to the specified reference position. By default, the reference position is the first position fix generated by the FirstGPS library. This feature is useful only when the MPM is stationary and its exact position is known.

Window Menu

The *Window* menu offers one item:

Always On Top

Select *Window / Always on Top* to have the GUI window always visible on the screen (never overlapped by other windows).

Help Menu

The *Help* menu offers two items:

Product Info

Select *Help / Product Info* to display product information. For more information, refer to ***navGetNavVersion***, page 69.

About

Select *Help / About* to display the *About* dialog with copyright and contact information.

Configuring FirstGPS Monitor for the Output Protocols

The FirstGPS Monitor supports NMEA, the industry-standard ASCII output protocol, which can be used through a serial interface from a PC COM port.

To configure the FirstGPS Monitor for NMEA output, you must specify the application port in the program settings using *Configure / Settings* from the main menu. The application port must be a COM port different from the one connected to the MPM unit. Set the application port parameters as desired. The default NMEA parameters are: 4800 baud, none, 8, 1.

NMEA outputs can be viewed on an ASCII terminal emulation program, such as Hyperterm. This program is typically used on a second computer. A single computer can be used provided that there are three COM ports (one for MPM, one for NMEA protocol, and one for the ASCII output program). An RS-232 null modem should be used for PC-to-PC communications.

CHAPTER

3

Software Integration

In this chapter:

- Overview
- Power-Up and Power-Down
- Polled vs. Event-Driven Data Collection
- Event vs. API Table

Overview

The FirstGPS software library runs as a library to a host application under a Real-Time Operating System (RTOS). This chapter discusses a few of the design implementation issues.

Power-Up and Power-Down

The FirstGPS software is provided as a static library. It must be linked with the user-defined application task (AppTask) to generate object code that can be loaded onto a target environment and run. It is the AppTask's responsibility to start up and shut down the FirstGPS engine as required. The two API functions ***navGpsStart***, page 80, and ***navGpsShutDown***, page 79, are provided for that purpose.

Before any other API calls can be made, the AppTask must call ***navGpsStart***, page 80, once to power up the GPS engine. In response to this call, two events are generated:

- NAV_POWER_ON_ACK – The first event is generated immediately after the call to ***navGpsStart*** and indicates that a request to power up has been received, and that the GPS engine is starting up.
- NAV_START_UP_COMPLETE – The second event is generated a few moments later to indicate that the FirstGPS library and the IO have successfully started and are running.This may take up to 3 seconds, depending on the hardware and RTOS.

After the second event has been received, the AppTask may use other FirstGPS API functions to retrieve library data. In addition, any API calls that configure library components (such as the receiver or filter configuration) must also be called at this time, if required.

When GPS operation is not required, the library can be shut down to decrease power consumption. To power down the library, the application must call ***navGpsShutDown***, page 79. In response to this call, the NAV_READY_TO_SHUTDOWN event is generated as soon as the library is completely shut down. It may take up to several seconds before this event is generated.

Polled vs. Event-Driven Data Collection

The FirstGPS library provides facilities for both polled and event-driven data collection.

- Polled data collection involves the AppTask periodically calling FirstGPS API functions to retrieve various data from the library (position fix, almanac status, satellite data, and so on).

  The AppTask assumes the responsibility of scheduling data collection. This type of data collection is useful when responding to manual requests for data, such as through a serial cable, keyboard, or touch-screen interface. Polled data is also useful when the AppTask has a timer-based scheduling mechanism to periodically collect, store, and/or process the data available through the API.

- Event-driven data collection allows the AppTask to quickly retrieve data through the API as soon as the data is generated by FirstGPS. The library generates an event whenever significant new data is available while the library is running. For example, every time the library computes a new position/velocity fix, a ***New Position Fix Available*** event is generated.

The library can generate several different types of events. The Standard FirstGPS Events table on page 58 lists all of them. Based on the type of event generated, the AppTask calls a corresponding API function to retrieve data associated with the event. The Event vs. API Table, page 35, lists FirstGPS events and the corresponding API function calls.

Whenever an event is generated, the event data structure is stored internally in a queue in the library. In order for the application task to know which events have been generated, a special API function, ***navGetEvent***, page 63, is used to retrieve FirstGPS events. This function blocks the calling task (for example, AppTask) until an event is generated at which point the task is resumed.

In a typical application, the AppTask runs in a simple loop, calling ***navGetEvent*** continuously. When an event is generated by the FirstGPS library, the function returns the event information, allowing the application task to call the API function corresponding to the generated event and process the data. After the event-specific processing is complete, the application task calls ***navGetEvent*** again and repeats the loop.

Note – *Event-driven data collection is the preferred method of using the FirstGPS API calls. Trimble recommends that polled data collection be used mainly when responding to queries for library data. For example, when a* ***New Position Fix Available*** *event is generated, the AppTask calls* ***navGetLastFix****, page 66, to retrieve the newly generated position fix data. The application should avoid scenarios where it constantly calls API functions, such as* ***navGetLastFix*** *on its own (scheduled by a timer or while running in an infinite loop).*

For information on how to retrieve events properly, refer to the description of the API function ***navGetEvent***, page 63 .

Event vs. API Table

EVENT DESCRIPTION	EVENT ID	API
New position/velocity fix available	NAV_FIX_NEW	navGetLastFix (page 66)
No new fix available	NAV_NO_FIX_NEW	n/a
New satellite tracking selection	NAV_TRK_SELECTION_NEW	navGetChStatus (page 60)
New satellite fix selection available	NAV_FIX_SV_SEL_NEW	navGetSvSelections (page 75)
New differential correction available	NAV_REF_DCORR_NEW	navGetDgpsCorrParams (page 62)
Acknowledge to a request to start GPS	NAV_POWER_ON_ACK	navGpsStart (page 80)
FirstGPS library successfully started	NAV_START_UP_COMPLETE	navGpsStart (page 80)
FirstGPS library successfully shut down	NAV_READY_TO_SHUTDOWN	navGpsShutDown (page 79)
FirstGPS library failed to initialize and start	NAV_START_UP_FAILED	navGpsStart (page 80)
New temperature measurement data	NAV_TEMP_MEAS	navGetTempParams (page 76)
FirstGPS library fatal error generated	NAV_FATAL_ERROR	n/a

APPENDIX

A

Using the FirstGPS API

In this chapter:

- Overview
- Examples
- FirstGPS API - Simple Data Types
- FirstGPS API - Navigation Data Types
- FirstGPS API - #define Directives and Enumerated Data Types
- FirstGPS API - Return Types
- FirstGPS API - Function Descriptions

Overview

This appendix describes the FirstGPS API data types, return types, function descriptions, and so on.

Examples

Refer to the sample application source code on the CD-ROM shipped with your Starter Kit. The examples illustrate how to use the API calls.

FirstGPS API - Simple Data Types

The table below describes all the simple data types defined within the Navigation Platform.

Defined Data Type	Description
S8	Signed 8 bits integer.
U8	Unsigned 8 bits integer.
S16	Signed 16 bits integer.
U16	Unsigned 16 bits integer.
S32	Signed 32 bits integer.
U32	Unsigned 32 bits integer.
FLT	4 bytes single-precision floating point.
DBL	8 bytes double-precision floating point.
BOOLEAN	1 byte Boolean value.

FirstGPS API - Navigation Data Types

The following subsections list all the data structures used for the Navigation platform. All bit field descriptions assume the convention that the least significant bit (LSB), the ***first*** bit, is bit 0.

npAllChStatusType

npAllChStatusType is used to report the tracking status of all channels.

Data Type	Variable	Description
U32	ulGpsTowMsec	GPS time of week in milliseconds.
npChStatusType	tChStatus[N_CHANNELS]	Individual satellite tracking status. For details, refer to npChStatusType in the table below.

npChStatusType

npChStatusType describes the tracking status for a channel.

Data Type	Variable	Description
U8	ucSvId	SV prn number.
U8	ucNavStatus	Describes navigation status. For the possible values and description, see Navigation Status Code, page 52.
U8	ucAmu5	SNR in units of 0.2 AMU per bit.
S16	sCno	SNR in units of CNO. The range is 0 to 47.
U16	usReserved	Reserved for future use.
U16	usMeasStatus	Describes measurement status. For all the possible values and descriptions, see Channel Status Code, page 52.
FLT	fltAzimuth	Azimuth in radians.
FLT	fltElev	Elevation in radians.

npCompTimeType

npCompTimeType defines a minimal set of clock/time fields that can be used to do conversions between GPS time and SCOUNT.

Data Type	Variable	Description
U32	ulMsecs	GPS time of week in milliseconds.
U16	usWeeks	GPS week number.
U16	usRefScount	SCOUNT value.
U8	ucStatus	Status of the time. For all the possible values and descriptions, refer to Time Status Code, page 57.

npDgpsCorrType

npDgpsCorrType describes the differential correction block.

Data Type	Variable	Description
U8	ucSvId	Satellite ID.
U8	ucIod	Issue of data.
DBL	dblTime	Correct time tag; GPS time of week in seconds.
U32	ulTimeOfReception	Time of reception; GPS time of week in milliseconds.
FLT	fltPrc	Pseudorange correction in meters.
FLT	fltRrc	Range rate correction in meters/second.
FLT	fltRrcDot	Acceleration correction in meters/second^2.
FLT	fltPrcErrEst	1-sigma pseudorange correction error estimate in meters.

npDopType

npDopType contains a set of DOP (dilution of precision) parameters.

Data Type	Variable	Description
FLT	fltPDOP	Position DOP.
FLT	fltHDOP	Horizontal DOP.
FLT	fltVDOP	Vertical DOP.
FLT	fltTDOP	Time DOP.

npErrorEstType

npErrorEstType stores the system estimate of error of the produced fix. These are 1-sigma values and are computed based on the expected error from each satellite and the geometry of the constellation in use.

Data Type	Variable	Description
FLT	flt3D	Total position error in meters.
FLT	fltHoriz	Horizontal position error in meters.
FLT	fltVert	Vertical position error in meters.
FLT	fltTime	Time bias error in meters.

npFixChType

npFixChType stores channel-dependent fix data.

Data Type	Variable	Description
FLT	fltReserved[N_CHANNELS+1]	Reserved for future use.
S16	sCno[N_CHANNELS]	CNO of SVs used in fix. The range is 0 to 47.
U8	ucSvUsed[N_CHANNELS]	ID of satellites used.
U8	ucIode[N_CHANNELS]	IODE of satellites used.
U8	ucAmu5[N_CHANNELS]	AMU*5 of SVs used in fix. The range is 0-16*5=80.

npFixSvSelType

npFixSvSelType describes the current constellation of satellites used in the current position fix.

Data Type	Variable	Description
U8	ucStatus	Satellite status code. For all the possible values and descriptions, refer to Satellite Status Code on page 57.
U8	ucDimMode	Operating dimension mode. For all the possible values and descriptions, refer to Operating Dimension on page 53.
U8	ucDgpsMode	Differential GPS mode. For all the possible values and descriptions, refer to, DGPS Operating Mode on page 52.
U8	ucSatCnt	Number of satellites in the constellation.
S8	cSvUsed [N_CHANNELS]	A list of SVs in the constellation.
npDopType	tDop	Dilution of precision numbers. For more information, refer to npDopType, page 41.

npFixType

npFixType is used to store fix data.

Data Type	Variable	Description
DBL	dblPosLLA[3]	[0]—Latitude in radians. The range is from –π/2 to +π/2. [1]—Longitude in radians. The range is from –π to +π. [2]—Altitude above Datum ellipsoid in meters.
FLT	fltVelEnu[3]	[0]—East. [1]—North. [2]—Up velocities in meters/second. The range is from 0.0 to +300.0.
npErrorEstType	tErrorEst	Solution 1-sigma error estimate numbers. For more information, refer to npErrorEstType, page 41.

npNavConfigType

npNavConfigType contains a set of parameters which maintains the configuration of a GPS receiver.

Data Type	Variable	Description
FLT	fltElevMask	Elevation mask. (Radians 0.0 – π/2).
FLT	fltAmuMask	AMU mask.
FLT	fltDopMask	DOP mask.
FLT	fltPdopSwitch	PDOP 2D/3D switch level.
FLT	fltMaxOscOffset	The search algorithm assumes the oscillator is +/- fltMaxOscOffset from nominal. The larger the value, the longer it takes to acquire. Units are df/f.
U8	ucOprtngDim	Operating dimension. For all the possible values, refer to Operating Dimension, page 53.
U8	ucDgpsCfg	Differential GPS configuration mode. For all the possible values, refer to DGPS Configuration Mode, page 53.
U8	ucDynamics	Receiver modes. For all the possible values, refer to Dynamic Mode Flag, page 53.
U8	ucDgpsAgeLlimit	DGPS correction age threshold in seconds.
U8	ucLpFixRate	RF sequencing rates.

npNavFixType

npNavFixType is used to store and report all pertinent information about the current fix.

Data Type	Variable	Description
npFixChType	tFixCh	Channel dependent fix information.
npFixType	tRawFix	Raw position in latitude, longitude, Datum alt.
npFixType	tFiltFix	Filtered position in latitude, longitude, Datum alt.
DBL	dblDatumToMslAlt	Altitude conversion from Datum to MSL.
DBL	dblPosECEF[3]	Raw ECEF position.
FLT	fltVelECEF[3]	Raw ECEF velocities.
DBL	dblOscBias	Oscillator bias in meters.
FLT	fltOscFreq	Oscillator frequency in meters/second.
DBL	dblTimeTag	GPS time of fix in seconds. The range is from 0 to 604799.
U16	usWeekTag	GPS week of fix.
npDopType	tDop	Dilution of precision numbers. For more information, refer to npDopType, page 41.
S16	sDatumIndex	0 = WGS84 = default.
U8	ucNumSvsUsed	Number of SVs used.
U8	ucStatus	Indicates the validity status of this fix. For more information, refer to Position Fix Status, page 55.
U8	ucSource	Indicates the source of this position fix. For more information, refer to Position Fix Source, page 54.
U8	ucDgpsFlag	Differential GPS status. For more information, refer to Position Fix DGPS Status, page 55.

npNavStatusType

npNavStatusType contains a set of status words for the following important data items:

- GPS position availability – Specifies the state of the GPS position solution, that is, whether any position is available for use by the receiver or its source, and whether it came from outside the receiver or was computed internally.
- GPS time availability – Specifies the state of the GPS time solution, that is, whether the time came from outside the receiver, from one satellite, or from a GPS fix.
- Receiver oscillator frequency offset availability – Specifies whether the receiver has an estimate of its oscillator frequency offset and where it came from.
- Memory – Specifies if any non-volatile memory was available when the receiver began the current start-up cycle.
- Ephemeris – Each bit in this variable represents the ephemeris status of a particular satellite (1-32). The bit is one (1) if the ephemeris is in memory and valid. Otherwise the bit is zero (0). This field can be used to determine quickly if the receiver needs ephemeris data.
- Almanac – Specifies the current status of the almanac.
  A field similar to the one described above is also available for the almanac to determine quickly if the receiver needs almanac data.

Data Type	Parameter	Description
U32	ulEphStatus	The ephemeris status from the navigation platform. The least significant bit (LSB) represents the ephemeris status for satellite 1, and the most significant bit (MSB) represents satellite 32. The value 1 indicates that the ephemeris is present and that it is valid; whereas 0 indicates that the ephemeris is not present or that it is invalid.
U32	ulAlmPageStatus	The almanac page status from the navigation platform. The LSB represents the ephemeris status for satellite 1, and the MSB represents satellite 32. The value 1 indicates that the almanac has been collected; whereas 0 indicates that the almanac has not been collected. The almanac page being collected does not mean that the satellite is healthy. See ulAlmHealthStatus below for health status information.
U32	ulAlmHealthStatus	The almanac health status from the navigation platform. The LSB represents the almanac status for satellite 1, and the MSB represents satellite 32. The value 1 indicates that the satellite is unhealthy; 0 indicates otherwise (the satellite is healthy or information has not been collected yet).
U8	ucPosStatus	The position status from the navigation platform. For all possible values, refer to Position Status Code, page 56.
U8	ucTimeStatus	The time status from the navigation platform. For all possible values, refer to Time Status Code, page 57.
U8	ucOscStatus	The oscillator status from the navigation platform. For all possible values, refer to Oscillator Status Code, page 54.
U8	ucMemStatus	The memory status from the navigation platform can be either zero or non-zero, where non-zero indicates that the memory contents are valid and zero indicates that the contents are invalid.
U8	ucAlmStatus	The almanac status from the navigation platform. For all possible values, refer to Almanac Status Code, page 50 .

npOscStatusType

npOscStatusType is a set of parameters describing the oscillator. The parameters, ***ucHaveBias*** and ***ucHaveFreqOffset***, are flags indicating the current condition of the oscillator. For all the possible values, refer to Oscillator Status Code, page 54.

Data Type	Parameter	Description
DBL	dblBias	Oscillator bias in meters.
DBL	dblFreqOffset	Oscillator frequency offset in meters/second.
FLT	fltBiasUnc	Oscillator bias uncertainty in meters.
FLT	fltFreqOffsetUnc	Oscillator frequency offset uncertainty in meters/second.
U8	ucHaveBias	Status flag indicating validity of bias. For all possible values, refer to Oscillator Status Code, page 54.
U8	ucHaveFreqOffset	Status flag indicating validity of drift. For all possible values, refer to Operating Dimension, page 53.

npTempMeasType

npTempMeasType is a set of temperature measurement parameters.

Data Type	Variable	Description
U8	ucMsecInterval	The number of milliseconds over which the temperature measurement was made.
U32	ulMeasMsec	The current GPS time when the temperature measurement was made.
U32	ulMeasCount	The measured TCO frequency (in counts/ucMsecInterval). This value varies with the temperature.

npVersionTagType

npVersionTagType lists a complete description of a product version.

Data Type	Variable	Description
S8	cName[18]	Product name.
U8	ucMajor	Major version number.
U8	ucMinor	Minor version number.
U8	ucBeta	Beta version. 0 if this is a final release.
U8	ucMonth	Month of release.
U8	ucDay	Day of release.
U16	usYear	Year of release.

npVersionType

npVersionType stores the version number for the measurement platform firmware, navigation platform API, navigation platform software, native RTOS, and native processor. For a detailed description, refer to ***npVersionTagType***, page 49.

Data Type	Variable	Description
npVersionTagType	tMpFwVer	Measurement platform version.
npVersionTagType	tNpApiVer	Navigation platform API version.
npVersionTagType	tNpSwVer	Navigation platform software version.
npVersionTagType	tRtosVer	Native RTOS version.
npVersionTagType	tCpuVer	Native processor version.

FirstGPS API - #define Directives and Enumerated Data Types

Event Handling Constants

Define	Value	Description
EVENT_NO_WAIT	0	For more information, refer to the lTimeOut parameter of navGetEvent, page 63.
EVENT_WAIT_FOREVER	-1	For more information, refer to the lTimeOut parameter of navGetEvent, page 63.
EVENT_GET_FIRST	0	For more information, refer to the ucGetFlag parameter of navGetEvent, page 63.
EVENT_GET_LAST	1	For more information, refer to the ucGetFlag parameter of navGetEvent, page 63.

Almanac Status Code

Define	Value	Description
NO_ALMANAC_DATA	0	There is no almanac data in the database.
HEALTH_PRESENT	1	Indicates almanac health page is collected.
ALM_PAGES_PRESENT	2	Almanac is complete and current.
UTC_PRESENT	4	Indicates UTC page is collected.
IONO_PRESENT	8	Indicates ionospheric page is collected.

For the almanac status flag, if the value is 0, then there is no almanac data in the receiver. This is the normal situation after a cold reset. It is possible that a complete almanac was not collected before the receiver was turned off after a cold reset.

When the value is non-zero, some almanac pages are present in the receiver.

A value of 15 indicates the desired condition, namely that the almanac is complete and current.

The four bit positions in the byte indicate the presence of certain data:

1. The bit with value 1 set means that the receiver has an almanac health page. This page defines which satellites are present in the current GPS constellation.
2. If the receiver has an almanac health page, and if it has an almanac page for each satellite which is declared present, then the bit with value 2 will be set.
3. If the receiver has the UTC page, then the bit with value 4 will be set.
4. If the receiver has the IONO page, then the bit with value 8 will be set.

Channel Status Code

Define	Value	Description
STATUS_RESET	0	Know nothing condition.
STATUS_TIME_KNOWN	1	MSEC ambiguity resolved.
STATUS_FREQ_LOCK	2	Delta phase + range.
STATUS_PHASE_LOCK	4	Phase, delta phase + range.
STATUS_LOCK_POINT_KNOWN	8	Resolved 1/2 cycle.
STATUS_FRESH_MEAS	16	At least one measure in the last measured period; measured period is the inverse of the measurement rate.
STATUS_GOOD_PARITY	32	At least 1 good parity result.
STATUS_VALID_MEAS	64	Measurement passes “goodness” checks.
STATUS_HEALTHY	128	Satellite data = healthy.

Navigation Status Code

Define	Value	Description
EPH_DECODED	1	Ephemeris has been decoded.
EPH_GOOD	2	Ephemeris is good.
EPH_USED_IN_FIX	16	Used in the current position fix
EPH_DGPS_CORR	32	DGPS correction is available

DGPS Operating Mode

Define	Value	Description
DGPS_OFF	0	Differential GPS is off.
DGPS_ONLY	1	Set to differential GPS only.
DGPS_AUTO_OFF	2	Auto mode off.
DGPS_AUTO_ON	3	Auto mode on.

DGPS Configuration Mode

Define	Value	Description
DGPS_CFG_OFF	0	Differential GPS is off.
DGPS_CFG_ONLY	1	Set to differential GPS only.
DGPS_CFG_AUTO	2	Auto mode.

Dynamic Mode Flag

Define	Value	Description
DYN_UNDEFINED	0	Undefined dynamics.
DYN_LAND	1	Land dynamics.
DYN_SEA	2	Sea dynamics.
DYN_AIR	3	Air dynamics.
DYN_STATIONARY	4	Stationary dynamics.
N_DYN	5	Total number of dynamic modes.

Operating Dimension

These definitions are used when setting the receiver's operation mode or reporting fix satellite selection.

Define	Value	Description
DIM_2D	0	2D mode of operation.
DIM_3D	1	3D mode of operation.
DIM_1SV	2	1SV mode of operation.
DIM_AUTO	3	Auto mode of operation.
DIM_DGPS_REF	4	DGPS mode of operation.
DIM_2D_CLK_HOLD	5	2D Clock Hold mode of operation.
DIM_OVERDET_CLK	6	Over-determined clock mode.

Oscillator Status Code

Status flag for the oscillator drift/bias used by the ***ucHaveFreqOffset*** and ***ucHaveBias*** fields in ***npOscStatusType***.

Define	Value	Description
HAOSC_UNKNOWN	0	No information.
HAOSC_OLD	1	Old.
HAOSC_FIX	2	Based on position/timing fix.
HAOSC_APPX	3	Approximate 1 SV solution.
HAOSC_MEAS	4	Have knowledge from raw measurement.

Position Fix Source

Source of the current position fix contained in the ***ucSource*** field of the ***navFixType*** structure.

Define	Value	Description
FIX_SOURCE_NONE	0	No position in memory.
FIX_SOURCE_INIT	1	Position came from outside receiver and is approximate.
FIX_SOURCE_ACCU	2	Position came from outside receiver and is accurate to within 100 meters.
FIX_SOURCE_2D	3	2D fix.
FIX_SOURCE_3D	4	3D fix.
FIX_SOURCE_OLD	5	Fix is considered old, but is trusted from selected satellites.

Position Fix Status

Status of the current position fix contained in the ***ucStatus*** field of the ***navFixType*** structure.

Define	Value	Description
FIX_STATUS_NONE	0	No position in memory.
FIX_STATUS_OLD	1	Fix is considered old, but is trusted from selected satellites.
FIX_STATUS_NEW	2	Fix is valid and new.

Position Fix DGPS Status

Indicates if the position fix is differentially corrected.

Define	Value	Description
FIX_GPS_NOT_CORRECTED	0	Fix is not differentially corrected.
FIX_DGPS_CORRECTED	1	Fix is differentially corrected.

Position Status Code

Status flag ***ucPosStatus*** has the following valid values. Only the values (4–7) and (0x80+(4–7)) should be considered to indicate that a valid position is available for output (navigation).

Define	Value	Description
HAVE_POS_NONE	0	No position in memory.
HAVE_POS_INIT	1	Position came from outside receiver and is approximate.
HAVE_POS_ACCU	3	Position came from outside receiver and is accurate to within 100 meters.
HAVE_POS_FIX2A	4	2D fix in auto 2D/3D mode.
HAVE_POS_FIX2M	5	2D fix in manual 2D mode.
HAVE_POS_FIX3A	6	3D fix in auto 2D/3D mode.
HAVE_POS_FIX3M	7	3D fix in manual 2D/3D mode.
HAVE_POS_DFIX2A	(HAVE_POS_FIX2A | 0x80)	Same as HAVE_POS_FIX2A but differential mode.
HAVE_POS_DFIX2M	(HAVE_POS_FIX2M | 0x80)	Same as HAVE_POS_FIX2M but differential mode.
HAVE_POS_DFIX3A	(HAVE_POS_FIX3A | 0x80)	Same as HAVE_POS_FIX3A but differential mode.
HAVE_POS_DFIX3M	(HAVE_POS_FIX3M | 0x80)	Same as HAVE_POS_FIX3M but differential mode.
HAVE_POS_OLD	8	Fix is considered old, but is trusted from selected satellites.
HAVE_POS_APPX	9	Fix is converging.
HAVE_POS_AVSV	10	Fix is converging.
HAVE_POS_FRIM	11	Fix is converging.
HAVE_POS_INTERNAL	12	Internal fix - not suitable for use.

Satellite Status Code

Define	Value	Description
SCODE_DOING_FIXES	0	Doing position fixes.
SCODE_GOOD_1SV	1	A usable SV is available.
SCODE_NEED_TIME	2	Need GPS time.
SCODE_PDOP_HIGH	3	PDOP is too high.
SCODE_BAD_1SV	4	The chosen SV is unusable.
SCODE_0SVS	5	No usable satellites.
SCODE_1SV	6	Only one usable satellite.
SCODE_2SVS	7	Only two usable satellites.
SCODE_3SVS	8	Only three usable satellites.
SCODE_DCORR_GEN	9	Differential corrections.
SCODE_OVERDET_CLK	10	Over-determined clock.

Time Status Code

The time status flag has the following valid states. Only the highest state should be considered valid for setting the application's real-time clock.

Define	Value	Description
T_STATUS_NONE	0	Time is unknown.
T_STATUS_PROBLEM	1	Error detected. Now being fixed.
T_STATUS_APPX	2	Time is good to a few seconds, from outside the receiver.
T_STATUS_SET	3	Time is good to ±10 msecs, from tracking one satellite.
T_STATUS_BIAS_KNOWN	4	Time is good to ±1 µs, from a position fix.

npTimeStatusType is the type that enumerates the above constants.

Standard FirstGPS Events

The following lists all of the standard events that the library could possibly generate. For example, the event ***NAV_FIX_NEW*** indicates that a new position fix has been generated.

Define	Value	Description
NAV_FIX_NEW	10	New solution fix.
NAV_NO_FIX_NEW	11	Incapable of producing new fix.
NAV_TRK_SELECTION_NEW	12	New SV tracking selection.
NAV_FIX_SV_SEL_NEW	13	New SV fix selection.
NAV_REF_DCORR_NEW	15	Newly generated differential correction.
NAV_POWER_ON_ACK	20	Received request to turn on the FirstGPS software.
NAV_START_UP_COMPLETE	21	FirstGPS software started. Status valid.
NAV_READY_TO_SHUTDOWN	22	FirstGPS software ended. User can shutdown the application.
NAV_START_UP_FAILED	25	FirstGPS software startup failed.
NAV_TEMP_MEAS	27	Temperature measurement updated.
NAV_FATAL_ERROR	28	FirstGPS software fatal error generated.[1]

[1]NAV_FATAL_ERROR indicates that a fatal error has occurred, and that the library has been shut down. To restart the library, call navGpsStart (page 80).

FirstGPS API - Return Types

The currently defined return types are:

Define	Value	Description
NAV_ERROR	-1	A function call failed.
NAV_OK	0	A function call is successful.
NAV_INCOMPLETE	1	Data retrieved is not complete.

npRetCode is the FirstGPS API function return type that enumerates the above values.

FirstGPS API - Function Descriptions

navGetBbAddr

Get starting BBRAM address.

Description

The function ***navGetBbAddr*** provides the calling application with the size and address location of the battery backed up memory (BBRAM) section during the power-up or power-down sequence. During power-up, it is called by the host application prior to calling the ***navGpsStart*** function.

Included Header File

#include “npStdIface.h”

Format

npRetCode navGetBbAddr (U8** ppucAddr, U32* pulSize);

OutPut Parameters

Data Type	Parameter	Description
U8**	ppucAddr	A pointer to a pointer which contains the BBRAM beginning address updated by the navigation platform.
U32*	pulSize	A pointer to U32 that has the size of the entire BBRAM in bytes. This field is also updated by the navigation platform.

Return Values

Value	Description
NAV_OK	The BBRAM address is retrieved successfully.
NAV_ERROR	The BBRAM address is not retrieved successfully.

navGetChStatus

Get satellite tracking status.

Description

This function provides information (azimuth, elevation, AMU, and so on) of the satellites currently being tracked from the navigation platform. The data structure, ***npAllChStatusType***, has two fields: GPS time and an array of ***npChStatusType[N_CHANNELS]***, with each entry corresponding to a channel. The data type, ***npChStatusType*** contains all the tracking information for a channel.

Included Header File

#include "npStdIface.h"

Format

npRetCode navGetChStatus (npAllChStatusType* ptAllChStatus);

Output Parameter

Data Type	Parameter	Description
npAllChStatusType*	ptAllChStatus	A pointer to an application structure which will be updated by the navigation platform. For descriptions of each sub-field, refer to npAllChStatusType, page 39 and npChStatusType, page 39.

Return Values

Value	Description
NAV_OK	The tracking information is retrieved successfully.
NAV_ERROR	The tracking information is not retrieved successfully.

navGetCompactTime

Get current time.

Description

The ***navGetCompactTime*** gets the current GPS week number and GPS time of week from the navigation platform. The GPS time and week number are computed based on the OS ticks elapsed since the last time reported from the measurement unit.

Included Header File

#include “npStdIface.h”

Format

npRetCode navGetCompactTime (npCompTimeType* ptCompTime);

Output Parameter

Data Type	Parameter	Description
npCompTimeType*	ptCompTime	A pointer to the npCompTimeType data structure which includes various representations of GPS time. Although this function does not return a value, check the ucStatus field of the ptCompTime to determine whether the ptCompTime contains a valid time or not.

Return Values

Value	Description
NAV_OK	The GPS time is retrieved successfully.
NAV_ERROR	The GPS time is not retrieved successfully.

navGetDgpsCorrParams

Get differential correction information.

Description

This function gets a copy of all the current differential correction parameters available from the navigation platform's database.

Included Header File

#include “npStdIface.h”

Format

npRetCode navGetDgpsCorrParams (npDgpsCorrType ptDgpsCorr[], U8* pucCount);

Output Parameters

Data Type	Parameter	Description
npDgpsCorrType	ptDgpsCorr[]	A pointer to an array which contains the results from the navigation platform. For complete details, refer to npDgpsCorrType, page 40.
U8*	pucCount	Indicates number of valid entries.

Return Values

Value	Description
NAV_OK	The differential correction parameters are available.
NAV_ERROR	The differential correction parameters are not available.

navGetEvent

Retrieve a FirstGPS event.

Description

The events are queued up internally inside the FirstGPS library. To retrieve the events in the FIFO order (First In First Out), pass ***EVENT_GET_FIRST*** for ***ucGetFlag***. Otherwise, use ***EVENT_GET_LAST***. This retrieves the most recent event.

If no events have been generated at the time when this function is called, the calling task will be suspended until an event has been generated. Pass the ID of the calling task in ***pvTaskId.*** This must be an RTOS-specific pointer to a type of the task control block used by the RTOS. This pointer will be cast to a task type relevant to the RTOS which will allow the FirstGPS library to suspend the calling task.

lTimeOut specifies the timeout value in milliseconds after which the suspended task will be resumed. If the task times out and no event has been generated yet, ***NAV_ERROR*** is returned.

If some event has been generated before this function was called, the event will be retrieved as specified by the ***ucGetFlag*** value and the function will return immediately. Otherwise, the function will suspend the calling task until either an event has been generated or the timeout (***lTimeOut***) expired.

If an event has been retrieved successfully, the function returns ***NAV_OK*** and updates ***pulEvent*** and ***pvEventData*** pointers with the retrieved event and its data.

Trimble recommends that the application have a task dedicated exclusively to retrieving and processing FirstGPS events. For example, the entry point of such a task would be:

```
void eventHandlerTask (void)
 {
  U32 ulEvent, ulEventData;

  +/ Loop forever...
  while (1)
  {
          // Retrieve the FirstGPS event
          if (navGetEvent (&gtEventHandlerTask,
                        &ulEvent,
                        &ulEventData,
                        EVENT_WAIT_FOREVER,
                        EVENT_GET_FIRST) == NAV_OK)
          {
               processEvent (ulEvent, ulEventData);
          }

          // Here, sleep a little letting other tasks to complete.
          // This is optional, and may not be required by the
         // application.
         Sleep (10);
  }
}
```

Note – *In the above example:*
(1) ***gtEventHandlerTask*** *is assumed to be a global variable of the task control block type relevant to a native RTOS.*
(2) The timeout value in ***navGetEvent()*** *is specified as* ***EVENT_WAIT_FOREVER****. This will suspend the event task until an event has been generated. Trimble recommends this method for retrieving events.*
(3) Using the method in (2), ***EVENT_GET_FIRST*** *should be used for the* ***ucGetFlag*** *parameter. This will retrieve the events in the order in which they were generated.*

Included Header File

#include “npStdIface.h”

Format

npRetCode navGetEvent (void* pvTaskId, U32* pulEvent, void* pvEventData, S32 lTimeOut, U8 ucGetFlag);

Input Parameters

Data Type	Parameter	Description
void*	pvTaskId	A pointer to a task control block of the calling task.
S32	lTimeOut	A timeout value.
U8	ucGetFlag	Indicates the event retrieval method.

Output Parameters

Data Type	Parameter	Description
U32*	pulEvent	A pointer to a variable to be updated with the event.
void*	pvEventData	A pointer to a variable to be updated with data associated with the event.

Return Values

Value	Description
NAV_OK	The event is retrieved successfully.
NAV_ERROR	The event is not retrieved successfully.

navGetLastFix

Get last position fix information.

Description

Gets a copy of the complete fix structure reflecting the last solution fix. This contains various items including LLA and ECEF position/velocity solutions, both raw and filtered, and their uncertainties, clock bias, clock offset, list of satellites used, **i**ssue **o**f **d**ata (**e**phemeris) (IODE), and different mode flags.

The ***NAV_FIX_NEW*** event will be generated when a new position fix is available.

Included Header File

#include “npStdIface.h”

Format

npRetCode navGetLastFix (npNavFixType* ptNavFix);

Output Parameter

Data Type	Parameter	Description
npNavFixType*	ptNavFix	A pointer to the npNavFixType data structure, which has various representations of navigation information, updated by the navigation platform. For detailed descriptions of each sub-field, refer to npNavFixType, page 45.

Return Values

Value	Description
NAV_OK	The last position fix information is retrieved successfully.
NAV_ERROR	The last position fix information is not retrieved successfully.

navGetNavConfig

Get receiver settings.

Description

This function gets a copy of the current GPS configuration.

Included Header File

#include “npStdIface.h”

Format

npRetCode navGetNavConfig (npNavConfigType* ptNavConfig);

Output Parameter

Data Type	Parameter	Description
npNavConfigType*	ptNavConfig	A copy of receiver configuration data to be updated by the navigation platform.

Return Values

Value	Description
NAV_OK	The receiver configuration parameters are available.
NAV_ERROR	The receiver configuration parameters are not available.

navGetNavStatus

Get navigation platform status.

Description

This function obtains a copy of the navigation status structure.

Included Header File

#include “npStdIface.h”

Format

npRetCode navGetNavStatus (npNavStatusType* ptNavStatus);

Output Parameter

Data Type	Parameter	Description
npNavStatusType*	ptNavStatus	A structure containing Navigation status parameters.

Return Values

Value	Description
NAV_OK	The Navigation status is obtained successfully from the navigation platform.
NAV_ERROR	The Navigation status is not obtained successfully from the navigation platform.

navGetNavVersion

Get navigation platform version.

Description

This function gets a copy of the software version for the measurement platform firmware, navigation platform API, navigation platform software, native RTOS, and native processor.

Included Header File

#include “npStdIface.h”

Format

npRetCode navGetNavVersion (npVersionType* ptVersion);

Output Parameter

Data Type	Parameter	Description
npVersionType*	ptVersion	A pointer to a version structure to be updated by the navigation platform. For detailed descriptions of each sub-field, refer to npVersionType, page 49.

Return Values

Value	Description
NAV_OK	The firmware version is retrieved successfully.
NAV_ERROR	The firmware version is not retrieved successfully.

navGetOscStatus

Get oscillator offset status.

Description

navGetOscStatus gets the navigation platform's current estimates of oscillator bias, frequency offset, and their uncertainties.

Included Header File

#include "npStdIface.h"

Format

npRetCode navGetOscStatus (npOscStatusType* ptOscStatus);

Output Parameter

Data Type	Parameter	Description
npOscStatusType*	ptOscStatus	Contains oscillator and frequency offset information. For a complete listing, refer to npOscStatusType, page 48.

Return Values

Value	Description
NAV_OK	The oscillator drift offset has been retrieved successfully.
NAV_ERROR	The oscillator drift offset has not been retrieved successfully.

navInitTime

Put current time.

Description

Imports an external GPS time into the navigation platform. It adjusts the local GPS time immediately, and the GPS time on the measurement platform through message passing. The GPS time is only permitted to override the current clock value if the current time has not been set (that is, no satellites are tracked). The primary purpose of this function is to reduce the time to the first fix.

Included Header File

#include “npStdIface.h”

Format

npRetCode navInitTime (U16 usGpsWn, U32 ulGpsTowMsec, FLT fltTimeAccuracy);

Input Parameters

Data Type	Parameter	Description
U16	usGpsWn	GPS week number into the navigation platform.
U32	ulGpsTowMsec	GPS time of week in millisecond into the navigation platform.
FLT	fltTimeAccuracy	Accuracy of input time in milliseconds, -1 if unknown.

Return Values

Value	Description
NAV_OK	The GPS time is imported successfully.
NAV_ERROR	The GPS time is not imported successfully.

navPutDgpsCorrParams

Put differential correction information.

Description

This function puts a copy of all the current differential correction parameters into the navigation platform's database.

Included Header File

#include “npStdIface.h”

Format

npRetCode navPutDgpsCorrParams (npDgpsCorrType ptDgpsCorr[], U8 ucCount, U8 ucClearFlag);

Input Parameters

Data Type	Parameter	Description
npDgpsCorrType*	ptDgpsCorr	A pointer to an array which contains the values to the navigation platform. For complete details, refer to npDgpsCorrType, page 40.
U8	ucCount	Indicates number of valid entries.
U8	ucClearFlag	Boolean: TRUE: clears all previous entries. FALSE: does not clear previous entries.

Return Values

Value	Description
NAV_OK	The differential correction parameters are imported successfully.
NAV_ERROR	The differential correction parameters are not imported successfully.

navPutNavConfig

Put receiver settings.

Description

This function imports a copy of the current GPS configuration into the navigation platform.

Included Header File

#include “npStdIface.h”

Format

npRetCode navPutNavConfig (npNavConfigType* ptNavConfig);

Input Parameter

Data Type	Parameter	Description
npNavConfigType*	ptNavConfig	A pointer to a new receiver configuration structure.

Return Values

Value	Description
NAV_OK	The receiver configuration parameters are imported successfully.
NAV_ERROR	The receiver configuration parameters are not imported successfully.

navPutOscFreqOffset

Put oscillator drift offset.

Description

navPutOscFreqOffset allows the user to override the startup value of the oscillator frequency offset. This is of particular importance to those systems that do not have battery backed-up RAM (BBRAM). The current oscillator offset can be obtained from the ***navGetOscStatus*** function.

Included Header File

#include “npStdIface.h”

Format

npRetCode navPutOscFreqOffset (DBL dblFreqOffset, FLT fltFreqSearch);

Input Parameters

Data Type	Parameter	Description
DBL	dblFreqOffset	New oscillator frequency offset will be imported into the navigation platform in (meters/second).
FLT	fltFreqSearch	The frequency to search around the dblFreqOffset in (meters/second).

Return Values

Value	Description
NAV_OK	The new oscillator frequency offset has been entered successfully.
NAV_ERROR	The new oscillator frequency offset has not been entered successfully.

navGetSvSelections

Get satellite selection.

Description

This function returns a list of satellites used for a position fix. The structure, ***npFixSvSelType***, has the current constellation of satellites used in doing a position fix. It also contains modes of operation flags and DOP values. Notice that a negative ***cSvUsed*** number indicates that an SV was not included for the position fix calculation because of poor integrity. For detailed descriptions, refer to ***npFixSvSelType***, page 42.

Included Header File

#include “npStdIface.h”

Format

npRetCode navGetSvSelections (npFixSvSelType* ptFixSvSel);

Output Parameter

Data Type	Parameter	Description
npFixSvSelType*	ptFixSvSel	A pointer to a data structure which includes number of satellites, DOPs, and modes of operation. Use the ucStatus field of the structure to determine the current status of the navigation platform. For details, refer to npFixSvSelType, page 42.

Return Values

Value	Description
NAV_OK	The npFixSvSelType structure has valid data.
NAV_ERROR	The npFixSvSelType structure does not have valid data.

navGetTempParams

Get temperature data.

Description

The function ***navGetTempParams*** gets the temperature data from the navigation platform. The data contain the following information:

- The number of milliseconds over which the temperature was made
- The time in ***ulMeasMsec*** when the navigation platform received the temperature measurement
- The temperature measurement

The temperature measurement is really the number of counts of an edge counter circuit on the measurement platform. The frequency of the counter will vary as a function of temperature.

The ***NAV_TEMP_MEAS*** event will be generated when the temperature measurement has been completed.

Included Header File

#include “npStdIface.h”

Format

npRetCode navGetTempParam (npTempMeasType* ptTempMeas);

Output Parameter

Data Type	Parameter	Description
npTempMeasType*	ptTempMeas	The npTempMeasType has all the temperature parameters for a measurement. For more information, refer to npTempMeasType, page 48.

Return Values

Value	Description
NAV_OK	A copy of temperature measurement data is retrieved successfully.
NAV_INCOMPLETE	The measurement requested is not complete.
NAV_ERROR	A copy of temperature measurement data is not retrieved successfully.

navStartTempMeas

Start temperature measurement.

Description

The function ***navStartTempMeas*** sends a message to the measurement platform to perform a temperature measurement over the number of milliseconds specified in the function call.

Included Header File

#include “npStdIface.h”

Format

npRetCode navStartTempMeas (U8 ucMsecInterval);

Input Parameter

Data Type	Parameter	Description
U8	ucMsecInterval	The number of milliseconds over which the temperature measurement is made.

Return Values

Value	Description
NAV_OK	The message is sent successfully to the measurement platform.
NAV_ERROR	The message is not sent successfully to the measurement platform.

navGpsShutDown

Shutdown navigation platform.

Description

The shutdown function causes an immediate clearing of all the resources allocated in the startup. It is important to call the shutdown function before removing the power from a receiver. Failure to do so causes a longer startup time on the subsequent start. It will take from 1 to 2 seconds to complete the shutdown.

Included Header File

#include “npStdIface.h”

Format

npRetCode navGpsShutDown (void);

Parameters

N/A

Return Values

Value	Description
NAV_OK	The shutdown of navigation platform is successful.
NAV_ERROR	The shutdown of navigation platform is not successful.

navGpsStart

Navigation platform startup.

Description

This is the entry point of the navigation platform. It creates all the necessary tasks and allocates all the global resources used within the navigation platform. It also has a set of input parameters, which allows an application task to enter the initial time and accuracy of that initial time, to reduce the time needed to determine the first fix.

Included Header File

#include “npStdIface.h”

Format

npRetCode navGpsStart (FLT fltTimeAccuracy, U32 ulGpsMsec, U16 usGpsWeek, U8 ucTimeStatus);

Input Parameters

Data Type	Parameter	Description
FLT	fltTimeAccuracy	Accuracy of time value input in milliseconds, –1 if unknown.
U32	ulGpgMsec	Approximate GPS time of week in milliseconds.
U16	usGpsWeek	GPS week number.
U8	ucTimeStatus	A flag indicates whether or not the above values are accurate. The value can be either TRUE, indicating that the time is accurate, or FALSE, indicating that the time is approximate. Ignore if fltTimeAccuracy is –1.

Return Values

Value	Description
NAV_OK	The initialization of navigation platform is successful.
NAV_ERROR	The initialization of navigation platform is not successful.

navPutInitPosition

Input initial LLA position.

Description

The ***navPutInitPosition*** function allows the user to enter a known position on startup to minimize the time needed for a first fix. The initial position is an array of doubles, in the order of latitude, longitude and altitude. The ***fltAccuFlag*** field determines whether or not the input parameters are accurate.

Included Header File

#include “npStdIface.h”

Format

npRetCode navPutInitPosition (DBL pdblPosLLA[ ], FLT fltAccuFlag);

Input Parameters

Data Type	Parameter	Description
DBL*	pdblPosLLA	Initial position from user. This is an array of three doubles, which has the following order: latitude, longitude, and altitude.
FLT	fltAccuFlag	Indicates the accuracy in meters. Value 0 means do not change, and all negative values indicate not accurate.

Return Values

Value	Description
NAV_OK	The initialization of position is successful.
NAV_ERROR	The initialization of position is not successful.

Glossary

This section defines technical terms and abbreviations used in this manual.

AMU	A Trimble term: Arbitrary Measurement Unit. The AMU is defined as: $20_{\log 10}$ (AMU) +22.8 = CNO where CNO = C/N_0, that is, the carrier-to-noise ratio.
ASIC	Application-specific integrated circuit.
ASCII	American Standards Committee for Information Interchange. This organization established a hexadecimal-to-printed character translation table used by all computers to store character information.
API	Application Programming Interface.
BBRAM	Battery Backed-up RAM. *See* Start / Stop GPS, page 20 and Load BBRAM on start, page 26.
CNO	Carrier-to-noise ratio. *See* AMU.
Differential Global Positioning System (DGPS)	GPS positions can be differentially corrected (DGPS) in real time (using radio telemetry) or through postprocessing (no radio telemetry required).

Dilution of precision (DOP)

A description of the purely geometrical contribution to the uncertainty in a position fix, given by the expression:
DOP = SQRT TRACE (A)
where: $A = (G^T G)^{-1}$ of G, the geometry matrix, corresponding to the measurement connection matrix for the instantaneous position solution (dependent on satellite-receiver geometry)[1].

The DOP factor depends on the parameters of the position-fix solution. Standard terms for GPS application are:

Geometric DOP – Three position coordinates plus clock offset in the solution.

Position DOP – Three coordinates.

Horizontal DOP – Two horizontal coordinates.

Vertical DOP – Height only.

Time DOP – Clock offset only.

[1] Refer to *Global Positioning System: Theory and Applications*, *Volume I*. Parkinson, Spilker, Axelrad, and Enge (1996), p. 413.

DSP

Digital Signal Processor.

DOP

Dilution of precision.

Earth-Centered-Earth-Fixed coordinates

The Cartesian coordinate system in which the X direction is the intersection of the prime meridian (Greenwich) with the equator. The vectors rotate with the earth. Z is the direction of the spin axis.

ECEF

See Earth-Centered-Earth-Fixed coordinates.

EMI

Electromagnetic interference.

Global Positioning System (GPS)
A system involving the use of satellite technology to calculate an accurate three-dimensional position. A GPS system consists of a space segment (up to 24 NAVSTAR satellites in six different orbits), the control segment (five monitor stations, one master control station and three upload stations), and the user segment (GPS receivers).

NAVSTAR satellites carry extremely accurate atomic clocks and broadcast coherent simultaneous signals.

Horizontal DOP (HDOP)
See Dilution of precision.

ICD-GPS-200
GPS Interface Control Document, ICD-GPS-200, IRN-200B-PR-00J, Rev. B-PR, U.S. Air Force, July 1, 1992. Search for this document on the Web site:

http://www.space.com.af.mil/usspace/gps_support/gps_documentation.htm.

IF
Intermediate frequency.

IODE
Issue of data (ephemeris).

LO
Local oscillator.

MPM
Measurement Platform Module.

NMEA
A standard established by the National Marine Electronics Association (NMEA), that defines electrical signals, data transmission protocol, timing, and sentence formats for communicating navigation data between marine navigation instruments.

PDOP
A measure of the quality of the GPS position. PDOP values below 7 are satisfactory (the ideal PDOP value is one, and values from 2 to 4 are good). *See* Dilution of precision.

PRN	Pseudo random noise.
PVT	Position-Velocity-Time.
Real-time	This term means to be performed immediately, not at some later time.
RFIC	Radio Frequency Integrated Circuit.
RTOS	Real-time Operating System.
SAW	Surface acoustic wave.
SV	Space vehicle, that is, GPS satellite.
TDOP	*See* Dilution of precision.
TTFF	Time to First Fix.
TCO	Temperature-Compensated Oscillator.
TCXO	Temperature-Compensated Crystal Oscillator.
UTC	Universal Time Coordinated; Universal Coordinated Time.
VDOP	*See* Dilution of precision.
XTAL	Crystal, as used in Figure 1.1 on page 5, for example.

Index

A

B

C

D

E

F

G

H

I

L

M

N

O

P

R

S

T

U

V

W

Reader Comment Form

FirstGPS Starter Kit User Guide — **March 2001**
P/N: 44326-00-ENG — **Revision A**

We appreciate your comments and suggestions for improving this publication. Contributors of particularly helpful evaluations will receive a thank-you gift.

I use the following Trimble product ____________________

for ____________________

Please circle a response for each of the statements below:

1 = Strongly Agree 2 = Agree 3 = Neutral 4 = Disagree 5 = Strongly Disagree

The manual is well organized.	1	2	3	4	5
I can find the information I want.	1	2	3	4	5
The information in the manual is accurate.	1	2	3	4	5
I can easily understand the instructions.	1	2	3	4	5
The manual contains enough examples.	1	2	3	4	5
The examples are appropriate and helpful.	1	2	3	4	5
The layout and format are attractive and useful.	1	2	3	4	5
The illustrations are clear and helpful.	1	2	3	4	5
The manual is:	too long	just right	too short		

Please answer the following questions:

Which sections do you use the most? ____________________

What do you like best about the manual? ____________________

____

What do you like least about the manual? ____________________

____

Optional

Name ____________________

Company ____________________

Address ____________________

____

Telephone ____________________ Fax ____________________

Please mail to Trimble Navigation Limited, 645 North Mary Avenue, P.O. Box 3642, Sunnyvale, CA 94088-3642. Alternatively, email your comments and suggestions to ReaderFeedback@trimble.com. All comments and suggestions become the property of Trimble Navigation Limited.