SLICE
GPS Accessory Kit
User's Manual

Table of Contents

DTS Support

SLICE systems are designed to be reliable and simple to operate. Should you need assistance, DTS has support engineers worldwide with extensive product knowledge and test experience ready to help via telephone, e-mail or on-site visits.

The best way to contact a DTS support engineer is to submit a request through the DTS Help Center web portal (support.dtsweb.com). You must be registered (support.dtsweb.com/registration) to submit a request (https://support.dtsweb.com/hc/en-us/requests/new). Registration also enables access to additional self-help resources and non-public support information.

This manual supports the following products:

13000-40740: GPS Accessory Kit

Introduction

The DTS GPS Accessory Kit brings accurate GPS Time and Location to DTS Data Acquisition. GPS location, time, and velocity information are provided through a standard NMEA data stream and a 1PPS time signal captured by DTS DAS channels. In post-processing, users can recover channel-to-channel synchronization as well as data alignment with location and network time.

This document describes the theory of operation of this Accessory Kit, examples of how to connect and use the kit, and information on the software suit functionality. This document assumes that the reader has familiarity with the following concepts:

- DTS DAS, SLICEWare and the SLICE ecosystem for arming, triggering, collecting data, visualizing data, and downloading all or portions of the data to a PC.
- General GPS receiver behavior and usage typical of commercial handheld units.
- GPS information as delivered via a 1PPS signal and serial NMEA sentences (see http://www.gpsinformation.org/dale/nmea.htm for a primer).
- Any third-party data visualization and analysis tools such as MATLAB or National Instruments DIAdem

Accessory Kit Overview

The GPS Accessory kit is depicted in Figure 2 below. It consists of a COTS Garmin 19X NMEA 0182 GPS receiver, a breakout cable to connect to a SLICE stack, and a software utility that eases the extraction of location and timing data. A detailed description and specifications of the Garmin GPS receiver and its mounting options can be found in the User Manual available from the Garmin website:

https://support.garmin.com/support/manuals/manuals.htm?partNo=010-01010-00&language=en&country=US.

Additional items in the kit include cables for different power supply options and a programing cable and programming software utility.

Figure 1: The SLICE GPS Accessory Kit

There are three main electrical connections in the breakout cable. These are:

- 1 Pulse-per-second (1PPS) precision time pulse, directly sampled on a DAS channel.
- NMEA serial data stream, directly sampled on a DAS channel.
- Power connection for the GPS receiver. (Options for connecting the power are discussed below in the GPS Accessory Kit Setup and Usage section.)

Since 1PPS and NMEA are directly sampled using normal DTS DAS channels, the DAS data must be post-processed to extract the location and timing information contained in the NMEA and 1PPS signals. The extracted data can then be used to align data from all channels. This post-processing can be accomplished using the DTS Location and Time Extraction Utility (an executable Windows program) or its equivalent MATLAB/Octave script provided through the DTS Help Center. The Utility is distributed as both C# and MATLAB source code (including C# project files) so that users can tailor its features to suit their requirements. Utility usage and information pertinent to its modification are detailed in later sections.

The Accessory Kit, its connections, and data post-processing are summarized in the block diagram in Figure 22.

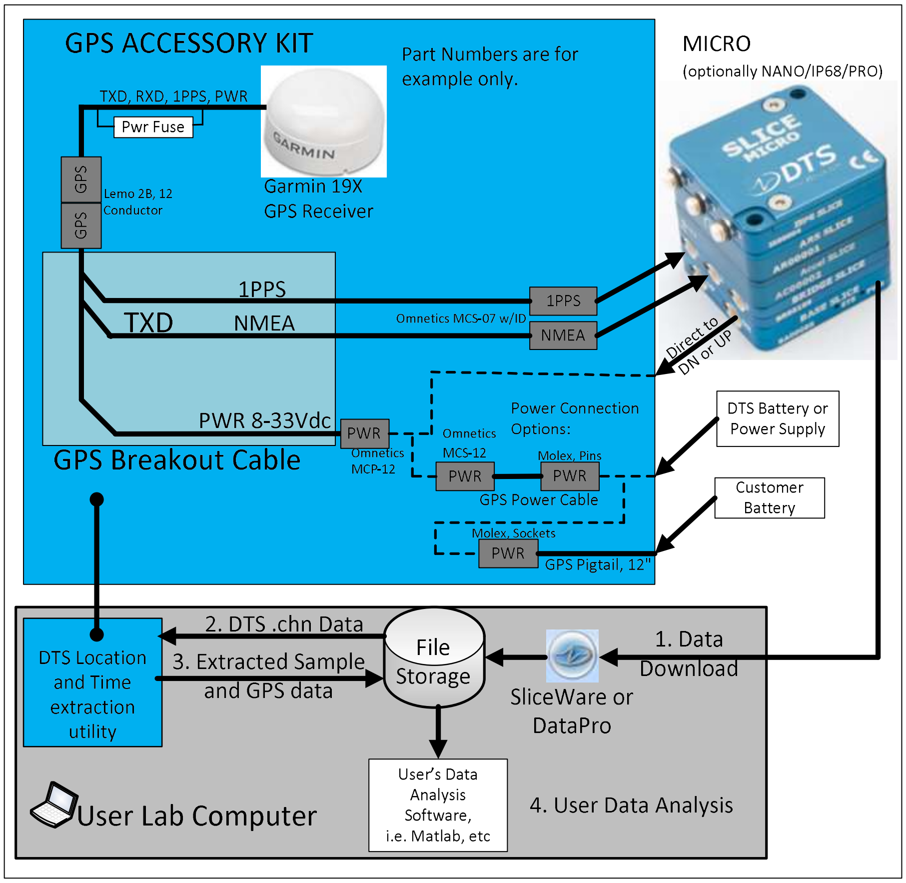

Figure 2: System Block Diagram, Hardware and Software

If multiple test locations are with equipped SLICE/GPS Accessory kits, all data can be aligned to common GPS location and time after data collection. This effectively allows precision sample-to-sample-to-GPS synchronization for all SLICE stacks, regardless of test location (**Error! Reference source not found.**2).

GPS Accessory Kit Setup and Usage

This section describes how to connect and use the system to align GPS time and location with captured data. Detailed information about connection schemes, signal descriptions, and configuration are discussed in later sections.

1. Connect the system as in Figure 3 and apply power.
    a. When communicating with SLICEWare to set up and configure the system, two power supplies, or an adapter from one power supply, will be needed
    b. When performing a test, and communication with the DAS is not required, power to the GPS receiver can be connected to the DAS UP or DN port or an external 12Vdc supply.
    c. **Error! Reference source not found.**3 and **Error! Reference source not found.**4 depict the interconnect for each of these configurations.

DTS Application Diagram
GPS Accessory Kit with MICRO/NANO Base Cable Kit

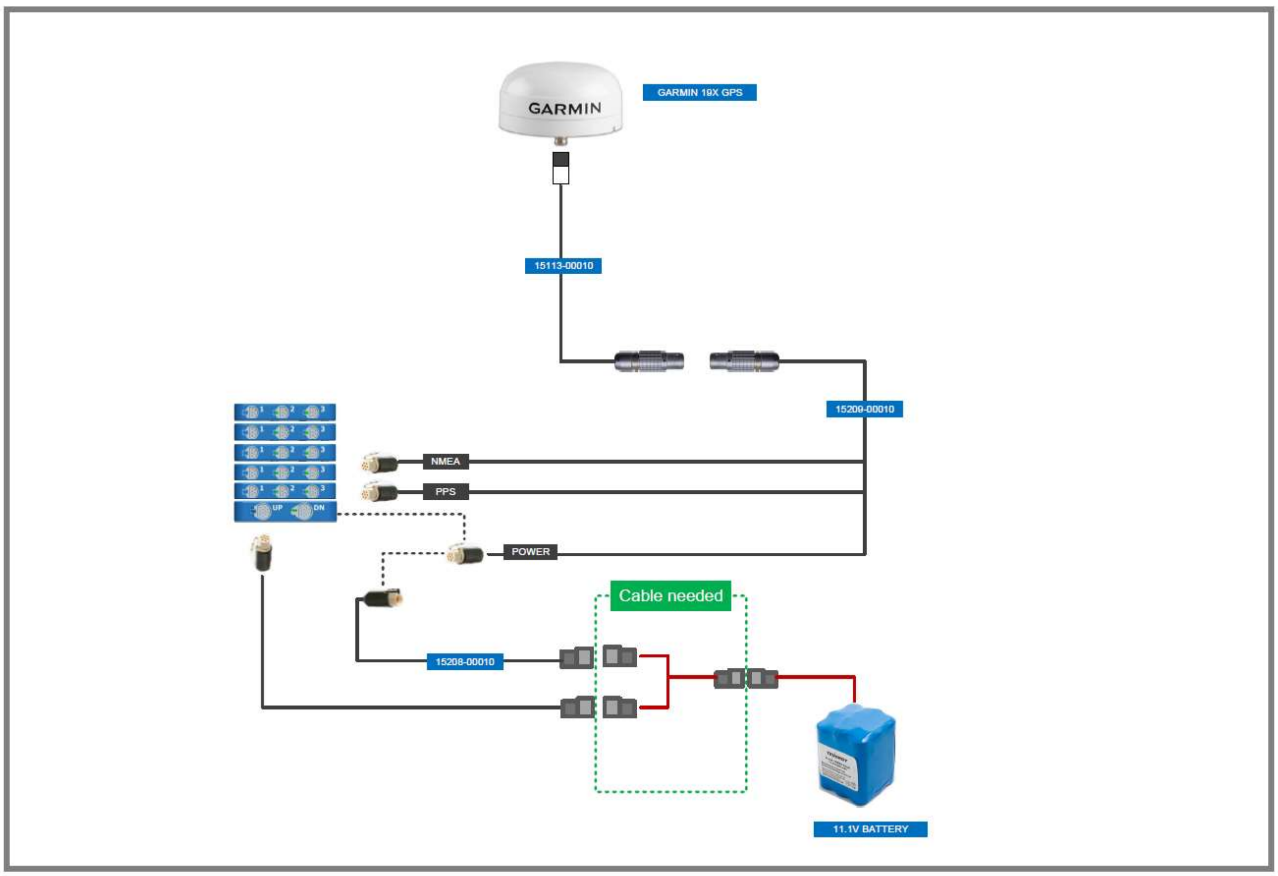

Figure 3: GPS Accessory Kit Connection – In Use, Powered from UP/DN or External Supply

Application Diagram
GPS Accessory Kit with MICRO/NANO Base Cable Kit

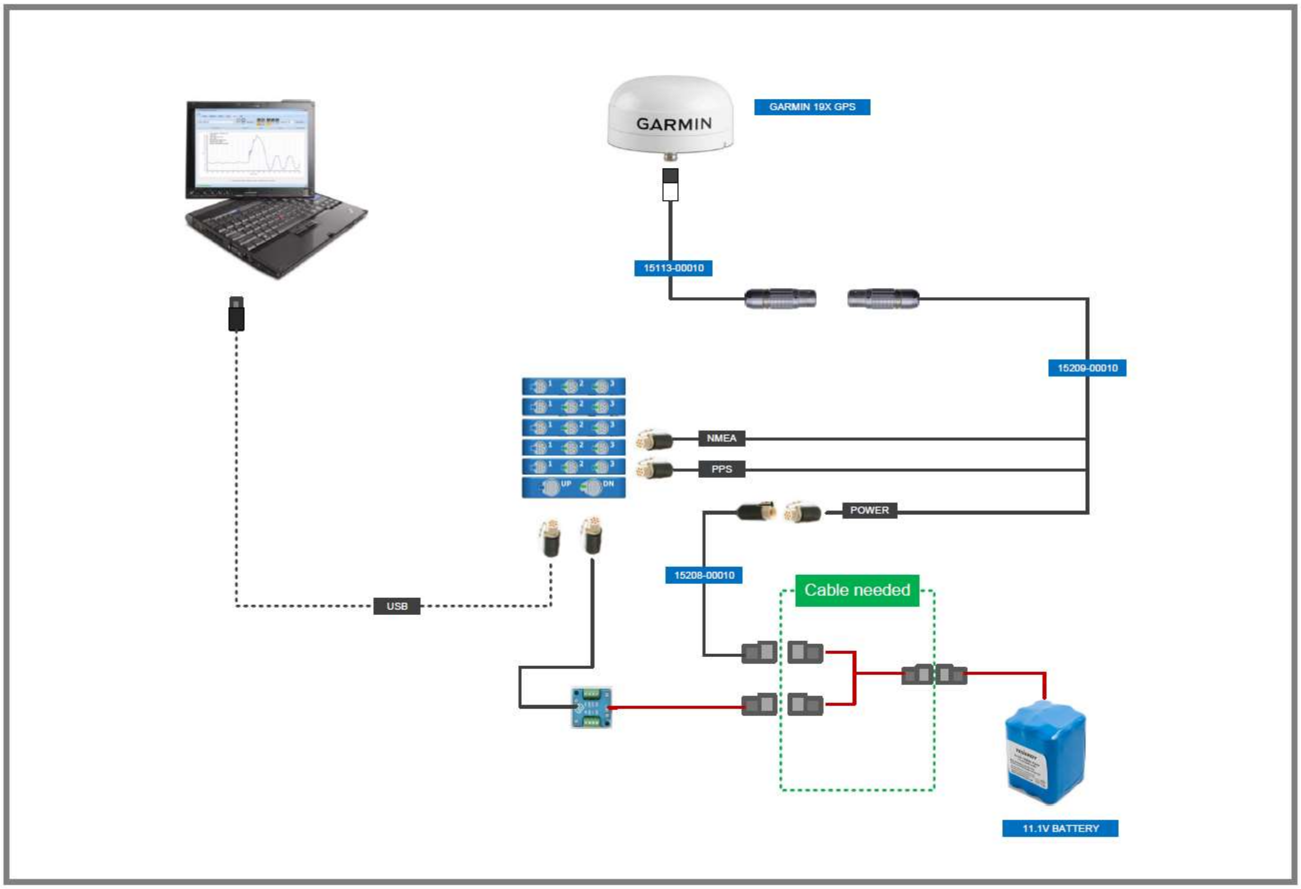

Figure 4: GPS Accessory Kit Connection – SLICEWare Configuration/Monitoring with EOC and External Supply

2. Allow the GPS receiver approximately 1 minute to acquire the GPS constellation and generate a navigation fix. This process can be faster in open sky environments. If the test environment blocks a clear view of the sky, it is recommended to collect several short samples of data and process it with the DTS Time and Location Extraction to ensure that GPS acquisition is possible.

3. Launch SLICEWare and import the GPS sensor settings provided as part of the documentation and code package. (See SLICEWare Manual for information on SLICEWare operation.) The figure below depicts the settings for the 1PPS and NMEA sensors.

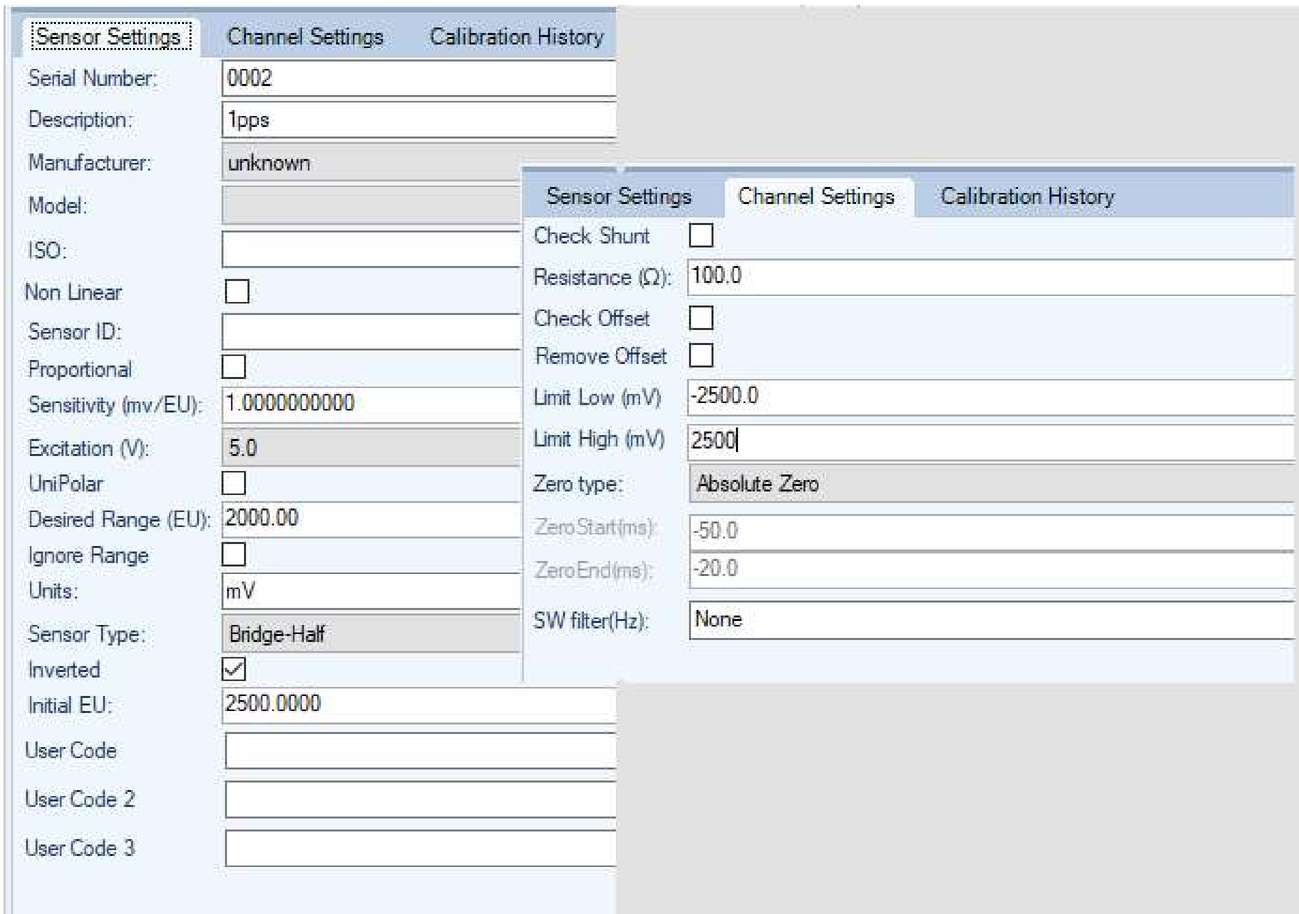

Figure 5: SLICEWare Sensor and Channel Settings

4. Configure and assign the 1PPS and NMEA inputs to the DAS as sensors to the following channel locations (Figure 6 below provides a screenshot of the position of these sensors in the SLICEWare Prepare tab):
    a. Channel 1 = 1PPS
    b. Channel 2 = NMEA

   The maximum sample rate must be set in the Prepare tab prior to moving on to the Diagnostics step.

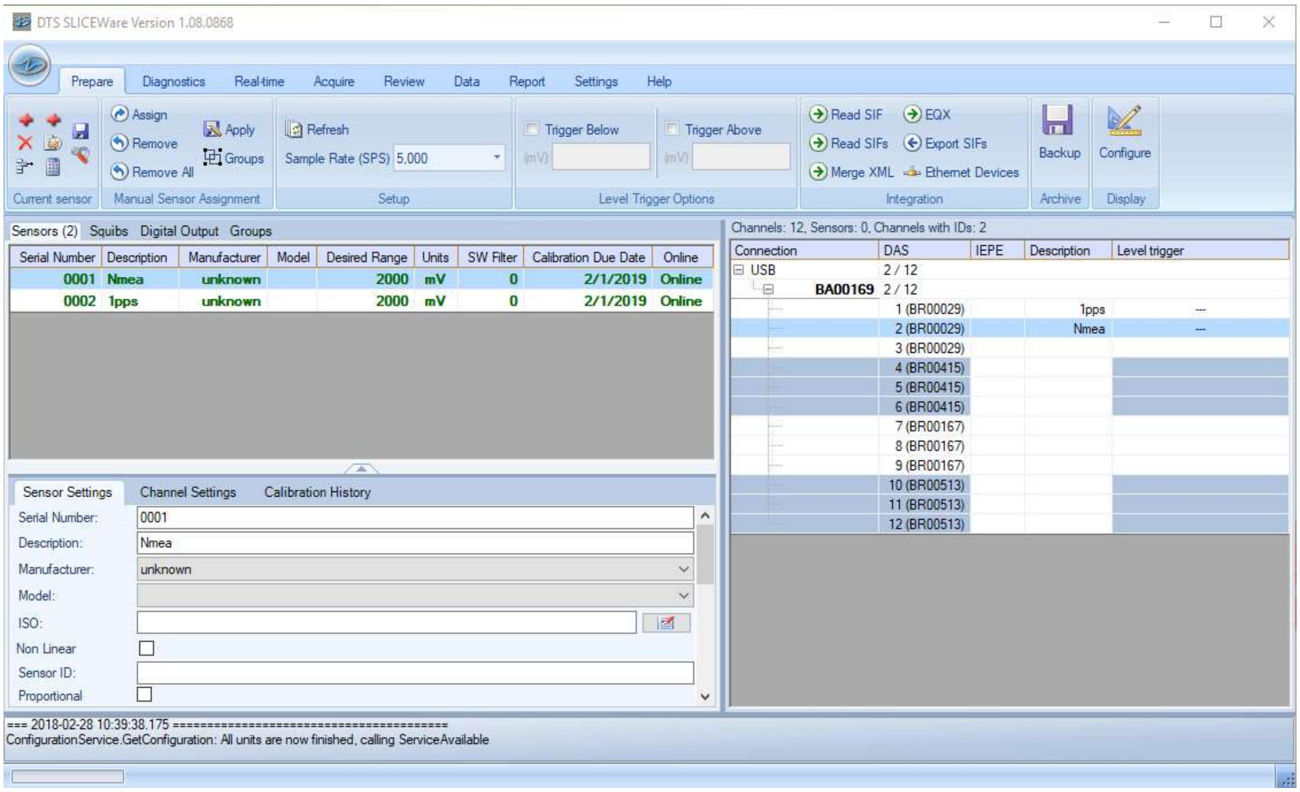

Figure 6: SLICEWare Sensor Channel Location

NOTE: The channel locations depicted in Figure 6, above, are achieved by connecting the GPS sensor inputs to the Channel 1 (1PPS) and Channel 2 (NMEA) inputs on the Bridge slice immediately above the Base slice in the DAS system.

5. Run sensor diagnostics before acquiring data. 1PPS and NMEA are active signals. As such, noise diagnostics may fail for these two channels; this is not indicative of a problem and diagnostics can be considered successful despite the channel failure(s).

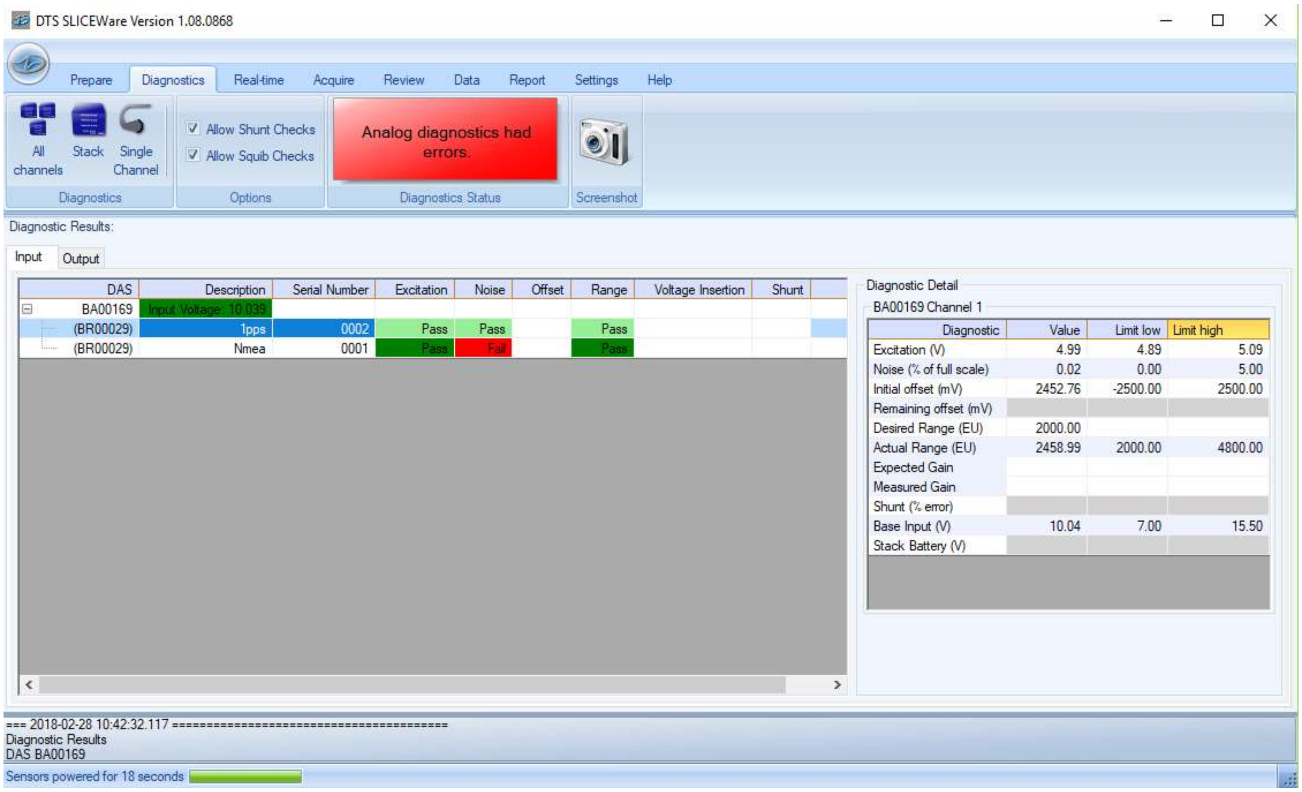

Figure 7: Diagnostics May Pass or Fail for the GPS Channels

6. Verify GPS activity in the SLICEWare Real-time tab. These signals should appear similar in magnitude and activity to the figure below. Real-time mode will undersample these signals, therefore the appearance of data in Real-time mode should not be used to determine proper functionality of the GPS receiver. A noisy signal in Real-time is not indicative of a GPS receiver issue.

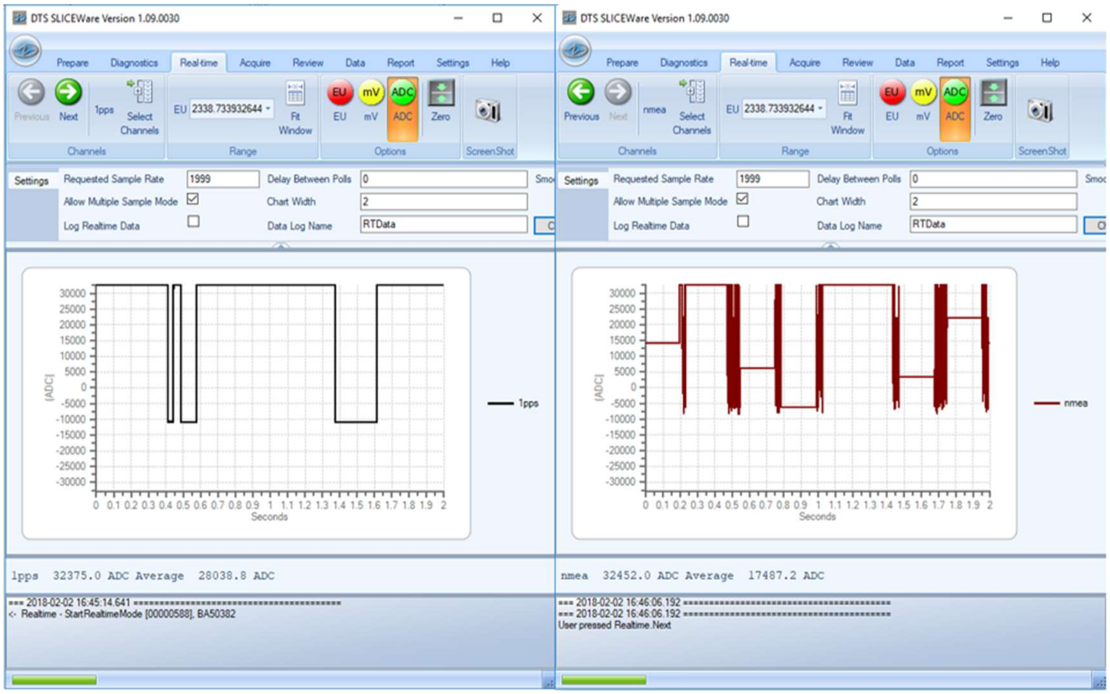

Figure 8: 1PPS and NMEA Signals in Real-time view

7. Confirm the sampling rate and recording parameters in the Acquire tab. Use SLICEWare to arm and collect data as normal.
    a. The factory default baud rate is 1200 which requires a sample rate of 5ksps or faster.

   NOTE: It is recommended to run a 30 second collection of data and process it using the DTS Time and Location Extraction Utility to ensure that the GPS receiver has a 'fix' before arming and collecting for a longer-term test. SLICEWare has no functionality to analyze 1PPS/NMEA data in real time and the appearance of these signals in the Real-time tab will be similar regardless of a proper GPS receiver fix.

8. After data collection is complete, retrieve the data from the DAS to the PC using SLICEWare. The default location for the collected data is C:\DTS\SLICEWare*[version]*\Data*[test id]*. The 1PPS data will be in the *.0.chn file, while the NMEA data will be in the *.1.chn file. Other sensor data from the remaining DAS channels will also be stored in this directory.

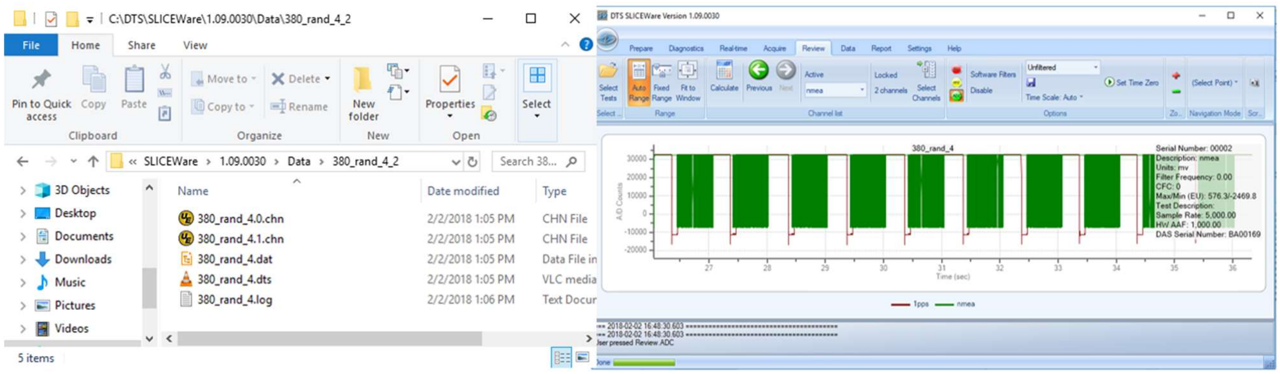

Figure 9: NMEA and 1PPS Visualized in SLICEWare and Data Directory

9. Run the Location and Time Extraction Utility as described in the Location and Time Extraction Utility and Usage section below.

Location and Time Extraction Utility and Usage

The DTS Location and Time Extraction Utility (or simply the "Utility") is used to parse through 1PPS and NMEA data collected as described above to extract location and time. The 1PPS and NMEA files can be parsed using either the Windows C# executable or the MATLAB/Octave script (hereafter referred to as the MATLAB script). All GPS data provided for processing must be:

- ➔ From the same data collection/test.
- ➔ Located in the same root directory.
- ➔ The same length (number of samples).
- ➔ More than 2 seconds in length, overall.

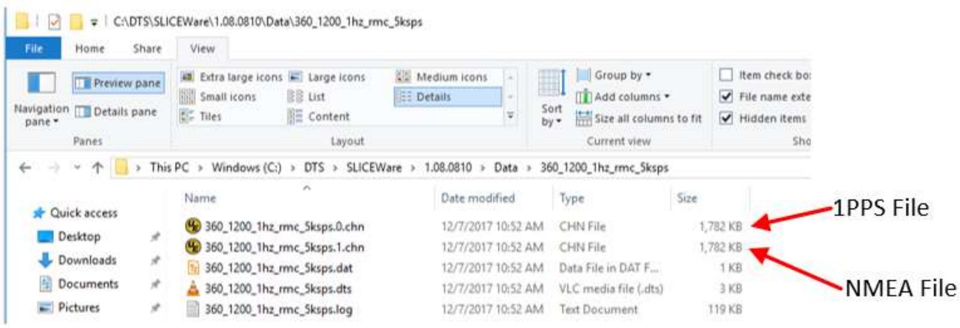

Figure 10: Source Directory Example for 1PPS and NMEA Data Files

All code and executable files are located on the supplied USB drive. The Windows application is included as both a binary executable and a Visual Studios 2017 C# project solution (*.sln).

- ➔ Windows Executable: [*USB*]:\Bin\setup.exe and GPSOne.application
- ➔ Windows C# Solution: [*USB*]:\Code\C# Solution\GPSOne.sln
- ➔ MATLAB: [*USB*]:\Code\Matlab

There are two processing modes available using the Utility: Summary and Extract.

Summary Mode

Summary mode examines a two second chunk of data at the beginning and end of the selected data file(s) and provides a location and time synopsis, dependent upon the combination of data files being processed. This mode is useful for getting a quick overview of location and time data for larger datasets without having to parse the entirety of the dataset. This mode generates a *_summary.txt file into the designated output directory.

Extract Mode

Extract mode parsed through the complete dataset and provides time, location, and summary files dependent upon the combination of data files being processed. These files will be output into the designated output directory. Dependent upon the length of the dataset being processed, the extract mode can take a significant amount of time and system processing power to execute.

Output File Types

Depending on the combination of 1PPS and NMEA data files being processed and the processing method being used, the Utility outputs different file types. For example, if both a 1PPS file and an NMEA file are used, the NMEA data is used to resolve the time of the 1PPS pulses; allowing for precision time and location data to be provided.

If only NMEA data is used, the timing of a sentence is estimated from the first rising edge of a valid NMEA burst; making location available with a coarse timing estimate attached for reference.

If only 1PPS data is used, the output provides the rising edge locations for each 1PPS edge within the dataset. Different combinations of data inputs yield different output file types. See Table 1 below for a summary of the output file names/types relative to the input dataset combinations.

Table 1: System Capability Depending on NMEA and 1PPS Availability

1PPS	NMEA	System Capability	Output Filenames
YES	YES	Location, Precision Network Time to Data Synch (<1 sample), Precision Stack-to-Stack Synch (<1 sample)	• *_summary.txt • *_extract.txt • *_nmea.nmea • *_time_estimate.csv • *_edge_location.csv • *_time.csv
NO	YES	Location, Coarse Network Time to Data Synch (<1s), Coarse Stack-to-Stack Synch (<1s)	• *_summary.txt • *_extract.txt • *_nmea.nmea • *_time.csv
YES	NO	Sample Frequency Accuracy (1ppm)	• *_summary.txt • *_extract.txt • *_nmea.nmea • *_time_estimate.csv
NO	NO	No Time or Location	• No output.

Table 2 provides a more detailed description of the output files.

Table 2: Output File Description

Output Filename	Data Description
*_summary.txt	Output from "Generate Summary", contains a brief overview of the first and last few seconds of the data record. Contains the location, time, and sample # of the first and last valid NMEA sentences, if NMEA is available. Contains the sample number where the first and last 1PPS rising edges occur, if 1PPS data is available. Example: ./test_v239/1pps_1nmea_summary/360_1200_1hz_rmc_5ksps Total Samples = 912042 Sample Rate = 5000 Test Duration = 0:3:2.4084 First 1PPS edge at sample # 2203 Last 1PPS edge at sample # 907220 First valid NMEA sentence begins at sample # 2543 UTC Time = 18:48:53 UTC Date = 07-12-17 Lat = 33 deg 45.4784' N Long = 118 deg 05.3443' W Last valid NMEA sentence begins at Sample # 907608 UTC Time = 18:51:54 UTC Date = 07-12-17 Lat = 33 deg 45.4781' N Long = 118 deg 05.3431' W

*_extract.txt	This is a summary that accompanies the output from the “Extract All” operation. It contains a summary of the full extraction data. Example: ./test_v239/1pps_1nmea_extract/360_1200_1hz_gga_5ksps Total Samples = 912042 Sample Rate = 5000 Test Duration = 0:3:2.4084 NMEA File Present Total complete NMEA bursts found = 181 Total valid NMEA RMC/GGA sentences= 181 1PPS File Present Total 1PPS edges used = 181
*_nmea.nmea	This file contains the extracted and decoded NMEA sentences as a human-readable text file. These sentences can be used by third party applications that understand NMEA text such as DIADem and various online utilities.
*_time_estimate.csv	This file is generated when only NMEA data is available. Contains columns that align the first rising edges of valid NMEA bursts with sample numbers in the DAS data record. Since the NMEA edge is being used for time and not the precision 1PPS pulse, the accuracy of the time can be as great as a few hundred milliseconds compared to UTC time.
*_edge_location.csv	This file is generated when only 1PPS data is available. It shows at which DAS data sample # a rising 1PPS edge is found.
*_time.csv	This file requires both 1PPS and NMEA data. It shows the DAS data sample #’s where precise time 1PPS rising edges occur and the associated NMEA time and location information.

Exceptional environmental conditions can cause corruption in the 1PPS and NMEA data. Examples might include extreme power glitches on the GPS system or DAS, cable interruptions, RF factors that affect the GPS radio, or other events. Figure 11 shows a MATLAB plot of the 1PPS and NMEA signals for two contrasting data records. The top record shows a properly captured dataset with no issues. The bottom case shows a recording where the NMEA cable was removed from the system and then replaced during operation.

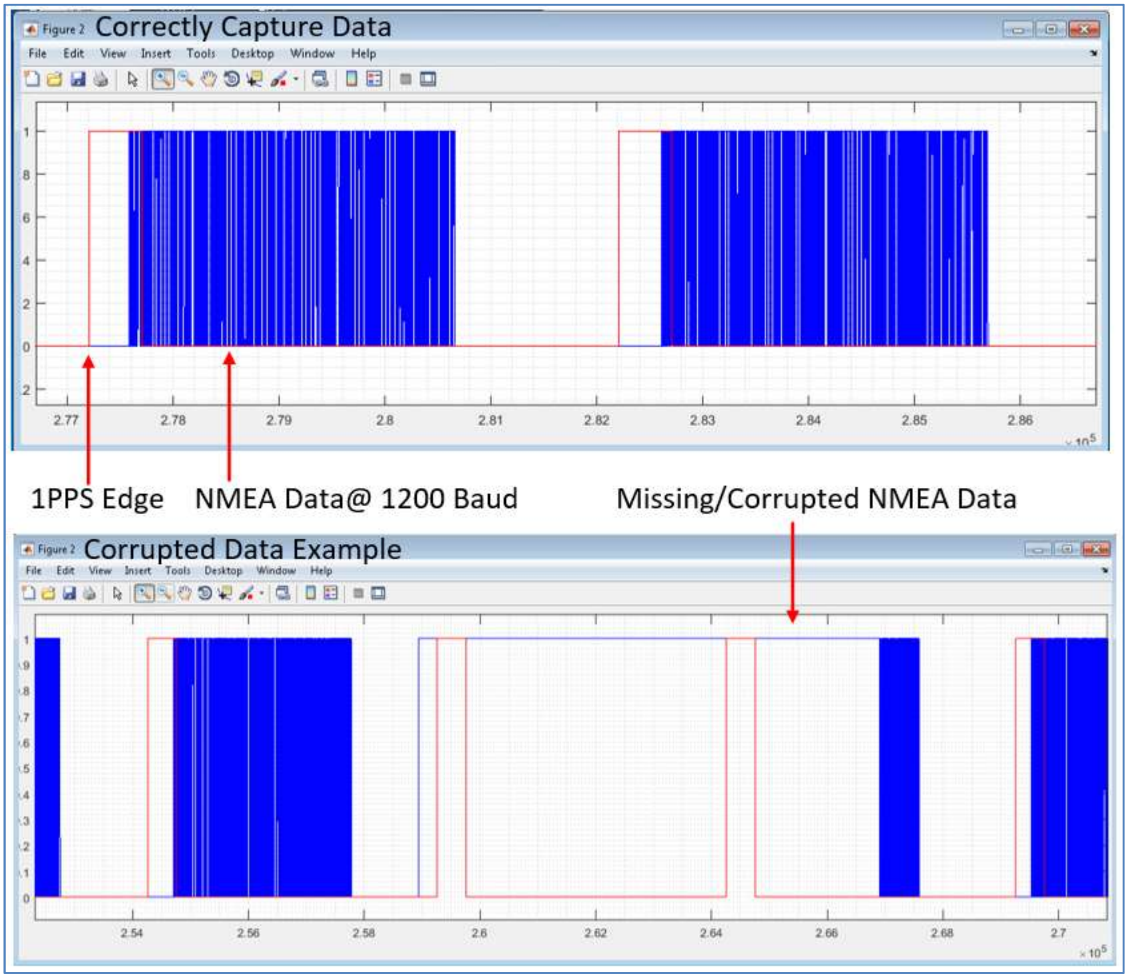

Figure 11: Comparison of Good and Bad NMEA/1PPS Data Visualized in MATLAB

The Utility attempts to counter these error events by tracking the number of 1PPS edges and valid NMEA bursts and notifying the user if the quantity does not match the total record length in seconds. This warning can be seen in the *_extract.txt output shown below. If dropouts occur, a number of utilities can be used to inspect the 1PPS and NMEA data for droupouts, including SLICEWare, MATLAB, and OCTAVE, among others. SLICEWare can be used to extract only the uncorrupted data of interest into *.chn files for processing with the Utility.

```
./test_v239/1pps_1nmea_extract/370_end_beg_drop_1
Total Samples = 912042
Sample Rate   = 5000
Test Duration = 0:3:2.4084
NMEA File Present
Total complete NMEA bursts found = 133
Total valid NMEA RMC/GGA sentences= 39
Warning, there may be dropouts or corruption in NMEA data
1PPS File Present
Total 1PPS edges used = 153
Warning, there may be dropouts in 1PPS data
```

Figure 12: An Extract Summary that Indicates Data Dropouts

Examples of test samples and outputs are provided on the USB drive in the [*DRIVE*]:\Code\Test Samples folder.

Windows C# Executable

The Windows C# Executable (hereafter referred to as the Windows Executable) provides the ability to process captured 1PPS and NMEA *.chn data in either single test or batch test modes. Invoking the Windows Executable is shown in Figure 13 below.

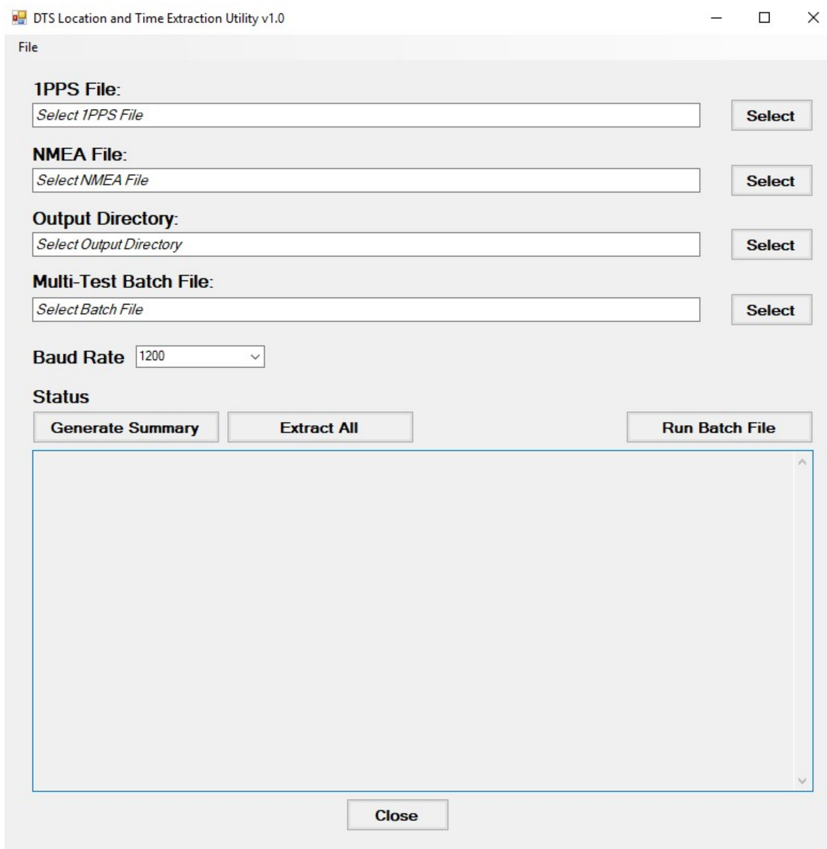

Figure 13: Windows Executable

Single Test Processing

Single test processing will process GPS data from individual 1PPS and/or NMEA files from a single data collection (test). The following steps can be used to process GPS data from a single test:

1. Select/Enter in the 1PPS and/or NMEA *.chn file path(s).
2. [Optional] Identify the Output Directory for the processed time and/or location files.
    a. If not output directory is identified, all generated files will be output to the folder where the 1PPS/NMEA *.chn file(s) are located.
3. Select the baud rate from the dropdown list.
    a. This should match the baud rate configured into the GPS receiver.
    b. The default baud rate the GPS Accessory kit is 1200.
4. Select “Generate Summary” or “Extract All”.

Batch File Processing

The Windows Executable batch file processing allows for the successive processing of multiple datasets at a time. To use the batch file functionality, a *.bat text file must be generated. Each line of the batch file runs a single dataset through the utility. The arguments for each line are as follows:

1. [OPTIONAL] The 1PPS file name and directory (*.0.chn data)

2. [OPTIONAL] The NMEA file name and directory (*.1.chn data)
3. Numerical Baud Rate (1200 (default), 2400, 4800, 9600, 19200, 38400)
4. Processing Method: 'summary' or 'extract'.
5. Desired Output Directory.
    a. If left empty (""), all generated output files will be placed in the root folder location for the GPS file(s) being processed.

```
".\data\test1.0.chn" ".\data\test1.1.chn" "1200" "summary"
".\testResults"

".\data\test2.0.chn" ".\data\test2.1.chn" "1200" "extract"
".\testResults"

"" ".\data\test3.1.chn" "1200" "summary" ".\testResults"

".\data\test3.0.chn" "" "4800" "extract" ".\testResults"
```

Figure 14 provides an example batch file that runs four datasets in succession. Notice that the Utility can process 1PPS data, NMEA data, or both.

NOTE: All entries must be enclosed by double quotes ("") to be considered valid by the Windows Executable Utility. Extraneous spaces between inputs will be ignored by the Utility, while extra lines between commands will result in and extra line between batch process summaries in the *testReport.txt report file (see below).


```
".\data\test1.0.chn" ".\data\test1.1.chn" "1200" "summary"
".\testResults"

".\data\test2.0.chn" ".\data\test2.1.chn" "1200" "extract"
".\testResults"

"" ".\data\test3.1.chn" "1200" "summary" ".\testResults"

".\data\test3.0.chn" "" "4800" "extract" ".\testResults"
```


Figure 14: FourTests.bat Batch File Example

Batch File processing is accomplished as follows:

1. Selected/Enter the path and file name of the *.bat batch file.
2. Select "Run Batch File".

A test report text file will be generated at the end of batch execution that will provide a summary of what processes were run (the command line in the batch file), a PASS/FAIL notice, a counter for the number of failed runs, and a summary of the errors (if any) for the given execution attempt. This file will be output to the same directory as the *.bat file that was supplied and will adhere to the following naming convention: [*.*bat file* name]testReport.txt.

Batch File Processing may require a significant amount of time to run. The amount of time required will be equal to the accumulated processing time for each of the datasets and processing methods that are listed in the *.bat file supplied.

MATLAB Script

The MATLAB script provides the same single-test functionality as the Windows Executable. At this time, there is no batch file functionality provided as part of the MATLAB script. While referred to as the MATLAB script, this code is also compatible with OCTAVE. Invocation of the MATLAB version of the Utility is as follows:

1. Open MATLAB and Navigate to the Directory where the Scripts Reside.
2. Call the Main Function (gpsMain) with the following arguments:
    a. The 1PPS file name and directory
    b. The NMEA file name and directory
    c. Numerical Baud Rate (1200 (default), 2400, 4800, 9600, 19200, 38400)
    d. Command: ‘summary’ or ‘extract’
    e. Desired Output Directory

The following are examples of calls to the main function to be entered into the MATLAB command prompt:

- >> gpsMain('./test0001.0.chn','./test0001.1.chn',1200,'summary','./')
- >> gpsMain('./test0001.0.chn','./test0001.1.chn',1200,'extract','./')

NOTE: All non-numerical entries must be enclosed by single quotes (‘’) to be considered valid. Extraneous white space is ignored.

An example run of the MATLAB script summary function is shown in Figure 15 below. A summary for the processed data is provided both in the form of an output *.txt file as well as being printed to the MATLAB command window for immediate viewing.

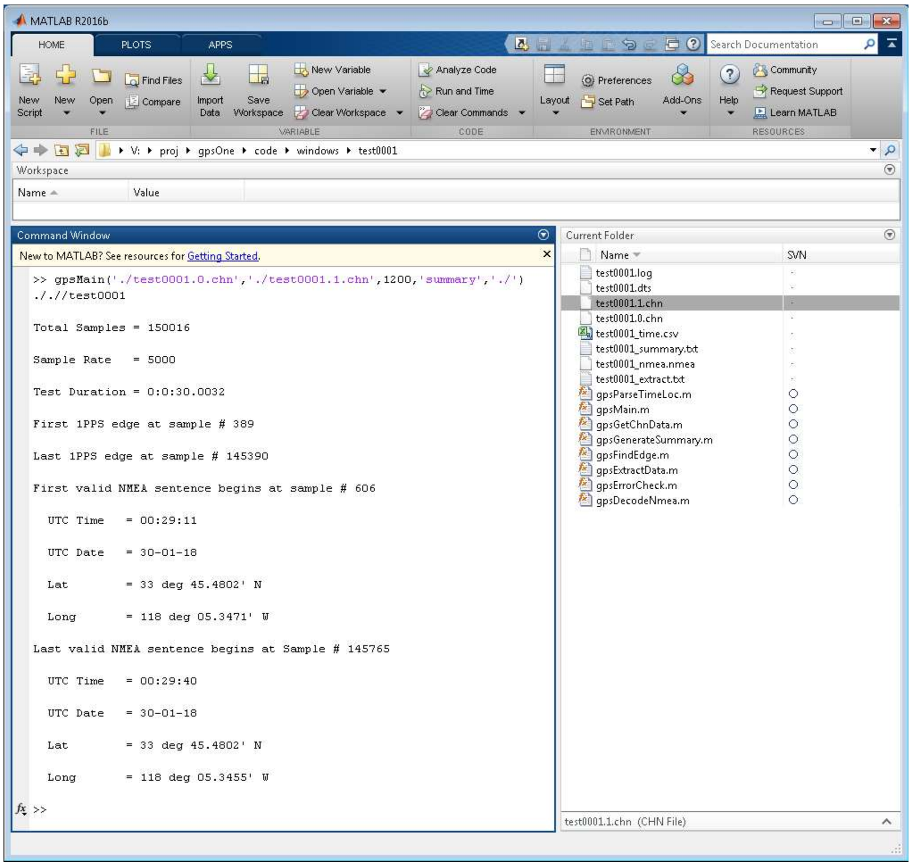

Figure 15: MATLAB Executable

Breakout Cable Connections

The GPS Accessory kit connections are summarized in

Table 5.

Table 3: GPS Accessory Kit Connections

Connection Name	Description and notes
1PPS	This is a 3.3V CMOS accurate timing pulse provided at 1 Pulse-per-second (1PPS). The rising edge is "on time" relative to Universal Coordinated Time (UTC) so long as the receiver has a valid navigation fix. On time accuracy of the edge is +/-1uS. The signal is used to indicate at what time any preceding navigations sentences are valid (see NMEA signal below). In the DTS system, this signal is directly sampled by a DAS channel. This allows timing accuracy down to +1 one sample period.
NMEA	This is a 3.3V CMOS ASCII serial data stream output at a frequency of 1 Hz. The serial data conforms to the National Maritime Electronics Association (NMEA) 0183 standard for delivering location, time, and GPS receiver data. At the factory, the Accessory Kit is configured to output the RMC (Recommended minimum specific GPS/Transit data) sentence at a baud rate of 1200. The RMC sentence provides UTC time and date, latitude and longitude, ground speed in knots, course made good, and magnetic variation. The data in the RMC sentence corresponds to the rising 1PPS edge that directly precedes it. This serial signal is directly sampled by the DAS channel and is decoded using a software utility detailed in the application section of this document. The factory configured baud rate and NMEA sentence can be reconfigured using the cable provided in the Accessory kit. See the NMEA and Baud Rate reconfiguration section of this document for Details.

Power	8-33Vdc power with a current draw of 40mA at 12Vdc. Power can be provided from several types of sources:
	■ Direct connection to the DAS UP/DN connector,
	■ Connection to DTS 12Vdc power supply (DTS P/N 13000-30541)
	■ 11.1V DTS Li-ion battery pack
	    ○ 2200mAh: DTS P/N 13000-30210
	    ○ 4400mAh: DTS P/N 13000-30220
	    ○ 6600mAh: DTS P/N 13000-30230
	    ○ Battery Charger DTS P/N 13000-30240
	■ User defined power source

Cable connections are depicted below.

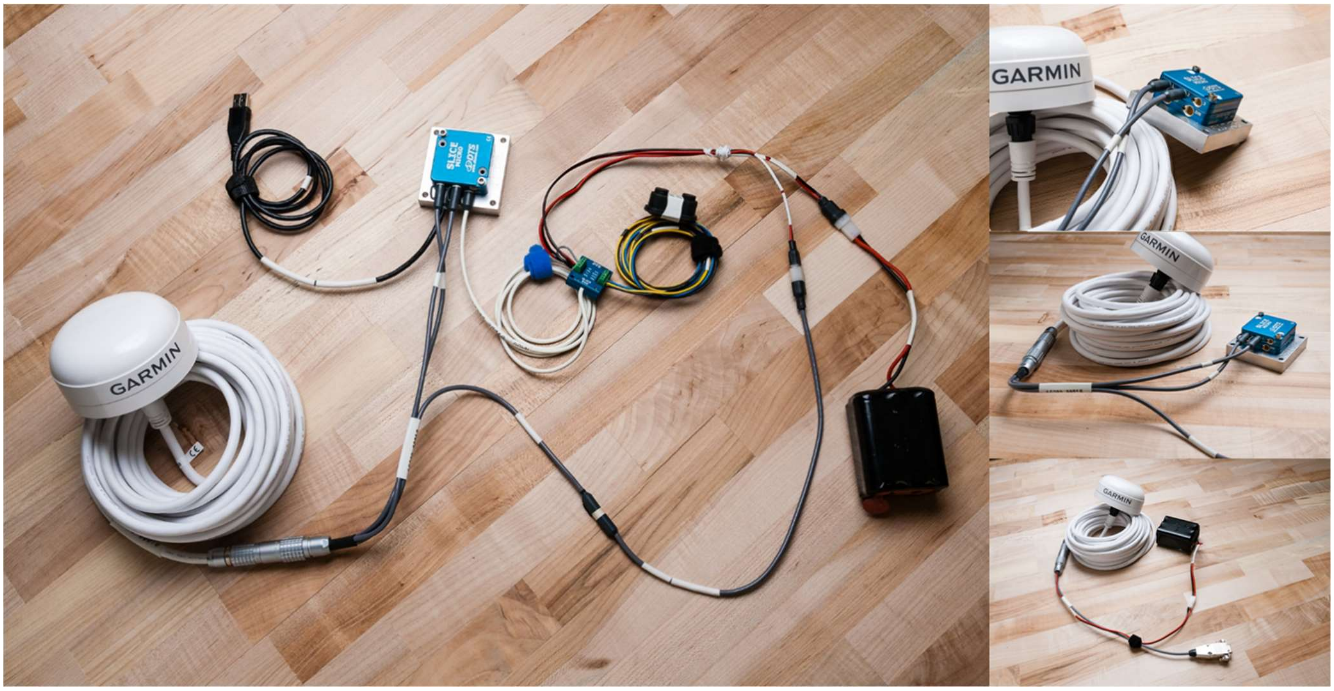

Figure 16: Setup and Connections

NMEA, Baud Rate, Sample Frequency and Receiver Configuration.

The GPS Accessory kit's factory default setting is the NMEA RMC sentence at 1200 baud (1200 bits-per-second). Since the NMEA data stream is directly sampled by a DAS channel, the DAS channel sampling frequency must be fast enough to reconstruct the data bits. The recommended sample rate is at least 4 times the baud rate. For example, a 1200 baud rate would have an appropriate sample rate for the DAS of 5ksps or higher. It is important to note that higher GPS baud rates require appropriately higher DAS sample rates.

Different NMEA sentences contain different information. For example, an RMC message does not indicate altitude whereas a GGA message does. This means that some measurement tasks may require different or even multiple NMEA sentences. One limitation is that all selected NMEA messages should complete transmission within the 1PPS interval. It may be necessary to increase the transmission baud rate as a result of increasing the number of NMEA sentences in order to successfully capture this data.

Table 5 shows recommended settings. Refer to the Garmin 19x HVS 0183 User Manual, Section 4.2, for information needed to estimate required baud rates vs NMEA sentences

(https://support.garmin.com/support/manuals/manuals.htm?partNo=010-01010-00&language=en&country=US).

Table 4: Example Recommended NMEA, Baud Rate, and Sample Frequency Settings

Baud	NMEA output Sentences	Minimum Recommended DAS Sample Frequency
1200	RMC or GGA	5ksps
4800	RMC and GGA	20ksps
4800	RMC, GGA, GSA, VTG	20ksps

Configuration of the NMEA sentence type, baud rate and DAS sample rate occurs in two places. The sample frequency is configured in SLICEWare prior to the DAS being armed and be adjusted before each data capture. Configuring NMEA sentence type and baud rate must be done using a special cable and software utility provided as part of the GPS accessory kit. Configuration of the NMEA and baud settings is persistent across power cycling of the receiver; requiring reconfiguration only upon changing GPS needs. The components required for this step are shown in

Table 5.

Table 5: Reconfiguration Components

Component	Description	Location
Accessory Kit Programming Cable	Connects the Garmin receiver to a PC serial port and a 12V power supply for programming.	DTS Part # 15207-0001x
Garmin SNSRXCFG v3.30	Software utility that communicates over a serial port to reconfigure the Garmin receiver. The Accessory Kit has been tested with version 3.30.	Provided with the GPS Accessory Kit support material and from Garmin here: http://www8.garmin.com/support/download_details.jsp?id=4053
PC Serial Port	Serial port capability on the user PC. This can be a hardware serial port, a generic USB-to-serial adapter, or the DTS USB-to-serial adapter DTS Part Number # 12000-00090.	DTS Part # 12000-00090.

Configuration Procedure: Example, RMC @ 1200 Baud

Follow this procedure to configure the GPS receiver to output the RMC NMEA message at 1200 Baud.

NOTE: This procedure applies to other NMEA and baud rate settings as well.

1. Download and install the SNSRXCFG Software Version 3.30 on a Windows PC or laptop. This Software can be found here:
   http://www8.garmin.com/support/download_details.jsp?id=4053
2. If a USB-to-serial adapter is used, connect the USB-to-Serial adapter to the programming PC. Use Window's Device Manager utility to locate the adapter communications port (COM5 in the example, Figure 17).

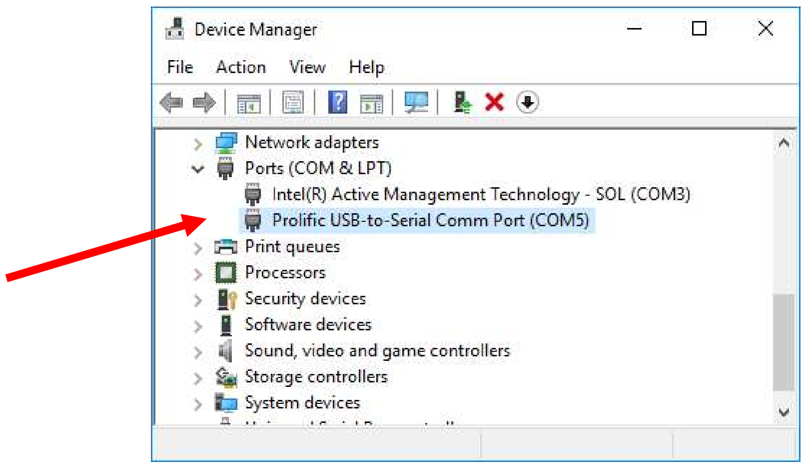

Figure 17: Windows Device Manager, USB-to-Serial Adapter COM5

3. Connect the GPS receiver as shown in Figure 18.
   a. Connect the Garmin receiver to the Accessory Kit Programming Cable.
   b. Connect the programming cable DB-9 connector to a hardware serial port or use a USB-to-Serial converter.

   NOTE: The blue USB-to-Serial converter depicted below (DTS 12000-00090) is not included in the GPS Accessory kit.

   c. Connect the power pigtail to a 12Vdc power supply.

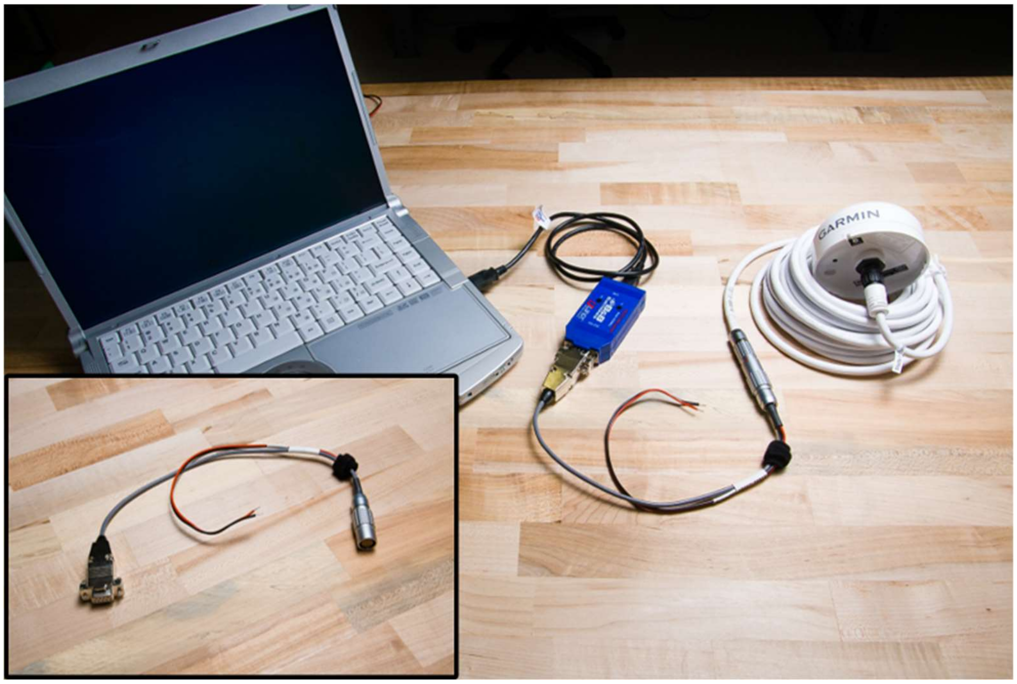

Figure 18: GPS Receiver Configuration Connection

4. Refer to the Garmin 19x HVS Technical Specifications, Appendix E for information on using SNSRXCFG to program the GPS receiver.
    a. Link to document: https://support.garmin.com/support/manuals/manuals.htm?partNo=010-01010-00&language=en&country=US
5. The programming sequence for RMC @ 1200 Baud can be summarized as in Figure 19:
    a. Set the COM port (COM5) in Comm→Setup and use the Auto Baud Rate selection.
    b. Click the Connect icon and allow the software utility to automatically connect to the GPS receiver.
    c. Set the baud rate to 1200 and the NMEA 2.30 Mode in Config→Sensor Configuration→Sensor Configuration Dialog Box. Hit OK to exit.
    d. Set the NMEA sentence to High Priority RMC and disable all other sentences. Click OK to exit.
    e. Click the Send Config to GPS button to upload the new configuration settings.
    f. Disconnect and cycle power to the receiver for the new settings to take effect.

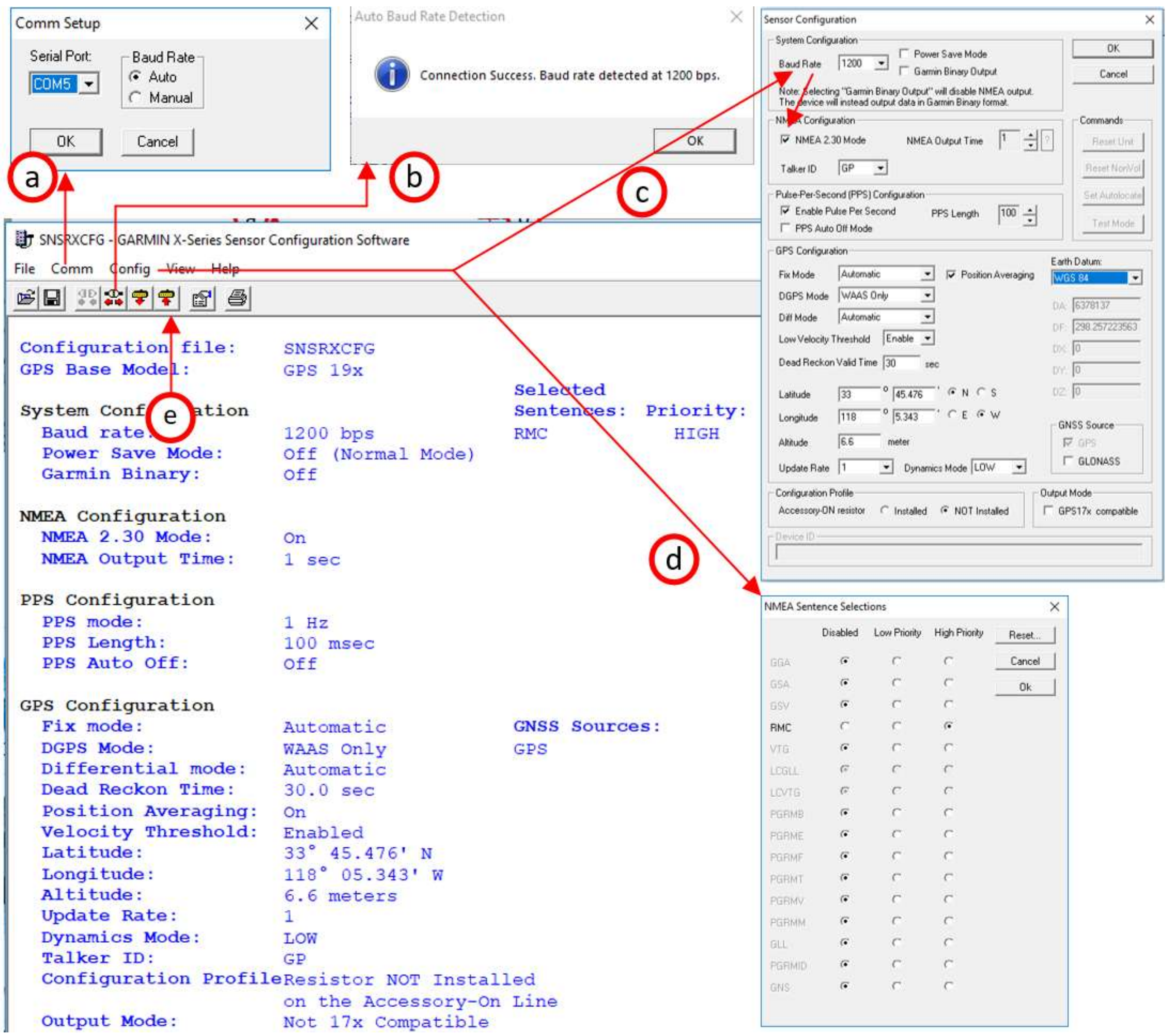

Figure 19: SNSRXCFG Usage Summary

Appendix A: Location and Time Extraction Utility Design Notes

The software is delivered in three formats:

- As a binary executable for the Windows operating system.
- As a Microsoft Visual Studio 2017 C# Source code solution
- MATLAB example scripts

The source code allows the user to modify functionality for desired features and performance or to integrate with the preferred data visualization tools such as MATLAB or National Instruments DIADem. This appendix provides an overview of the software architecture to assist user modification. Figure 20 shows how the main functions work together. Function descriptions follow in later sections.

NOTE: DTS is not responsible for any modifications made to the source code.

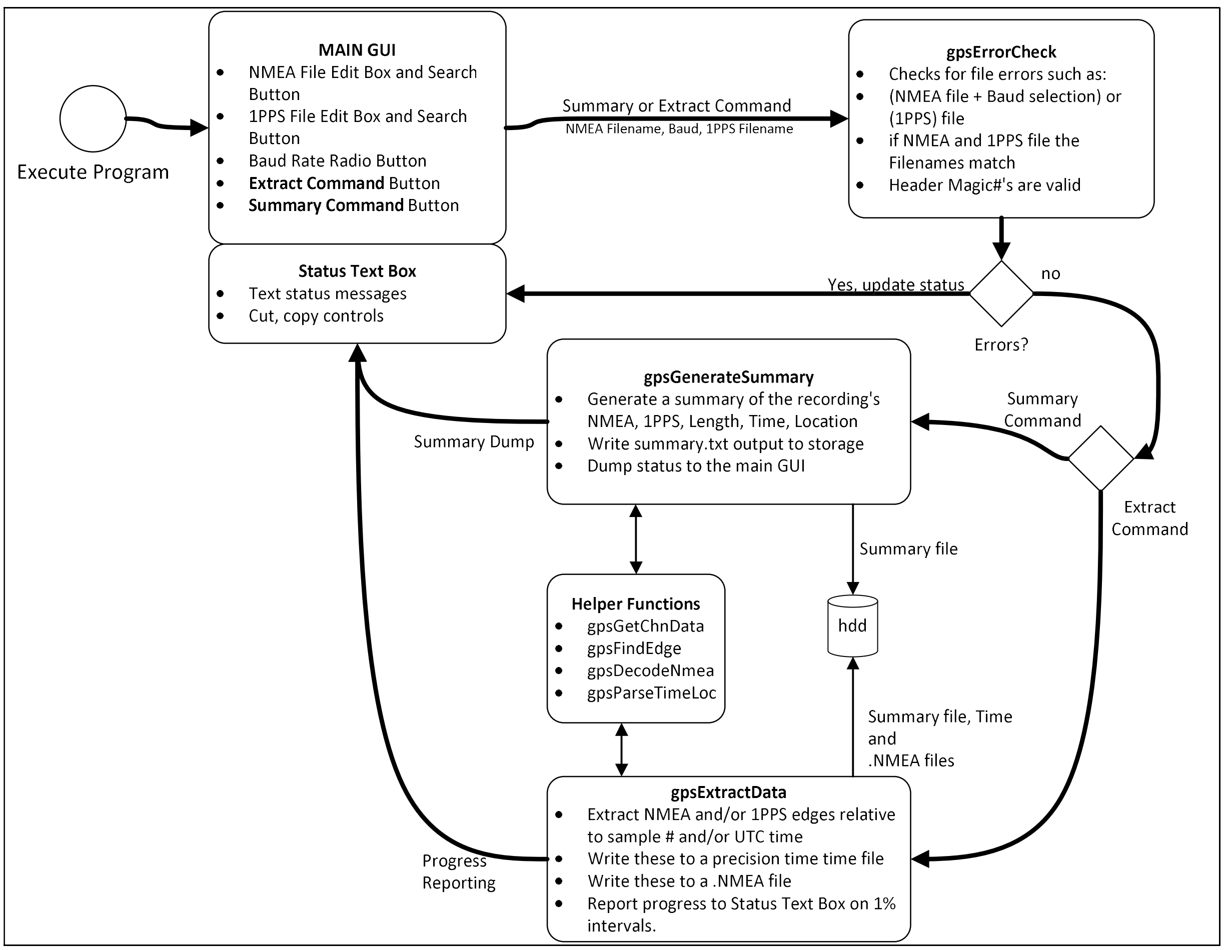

Figure 20: Program Flow

In the block diagram, the MAIN GUI represents either the C# executable or the MATLAB command interface. The gpsGenerateSummary examines only the first and last few seconds of any dataset to quickly deliver an overview of time and location, even when the dataset is very large. The gpsExtractData parses through an entire dataset and delivers the time and location information found; this process can be time and memory-intensive for very large datasets.

The functions gpsGenerateSummary and gpsExtractData rely on several lower level helper functions. In the MATLAB code these are captured in the following scripts. Key concepts to understanding how these functions work can be found in the following subsections.

1. gpsErrorCheck: Error checking of the 1PPS and/or NMEA data files to ensure they can be used to generate uncorrupted data. See the function header for a detailed description of which errors are screened.
2. gpsGetChnData: How the data is read into memory and prepared for analysis
3. gpsFindEdge: How a 1PPS or NMEA edge is detected in the data
4. gpsDecodeNmea: How NMEA data is recovered and decoded from the imported channel data.

For each of the following subsections, read the function header and follow along in the source code for more details.

Function gpsGetChnData

This function reads DAS *.chn data into memory and then filters the analog 1PPS and NMEA *.chn sample data into values of either 1 or 0. The DAS sample data consists of analog values with a typical mean of around 7500. This mean of 7500 was measured empirically over many test cycles. The filter algorithm is as follows:

1. Calculate the mean value of the 1PPS or NMEA data based on channel data values and then add the minimum value to the result. This allows for adjustment of the mean value relative to the minimum value.

$$calcMean = \frac{max + min}{2} + min$$

2. If this calculated mean value is significantly different (+/-20%) from the empirical mean (7500), this could indicate a marginal cable connection. In that case, the empirical mean is used for filtering. Otherwise, use the calculated mean value for filtering.

```
if ( (calcMean < (empiricalMean * 0.8)) || (calcMean > (empiricalMean * 1.2)) )
    meanVal = empiricalMean;
else
    meanVal = calcMean;
```

3. Parse through the entire dataset. Due to the use of a differential amplifier in the DAS channel inputs, the 1PPS and NMEA data channels come in on the -SIG line; meaning that the values of the signal need to be inverted to match the actual output from the GPS receiver.

Any sample value that is greater than the mean value (meanVal) is assigned a value of 0. Otherwise, a value of 1 is assigned.

$$
\begin{aligned}
&\mathit{if}\,(\,adcVal > meanVal\,)\\
&\quad adcVal = 0;\\
&\mathit{else}\\
&\quad adcVal = 1;
\end{aligned}
$$

Function gpsFindEdge

This function scans through the DAS data from beginning to end and finds the sample numbers at which positive edges occur. It returns these sample numbers as an array. This array can later be used to determine where parsing of an NMEA sentence burst begins or where a 1PPS edge is located. The algorithm to detect an edge is:

1. Determine the number of samples needed to hold 15 ASCII symbols (i.e. a boxcar filter)

$$Width = \left(\frac{5ksps\ sample\ frequency}{1200\ baud\ rate}\right) \times 15\ symbols = 63\ samples\ at\ a\ time$$

2. Starting at the beginning of the record, scan through the data and examine 63 samples at a time for the presence of an edge transition. An edge transition occurs when samples 1- 62 are a value of 0 and sample 63 is a value of 1. If this pattern is detected, mark the sample number location as a rising edge. Scanning is accomplished by right-shifting the boxcar by one sample at a time.

Function gpsDecodeNmea

The NMEA data is serial data with 8 data bits, no parity bits, 1 start bit, and 1 stop bit. It is sent at a specific baud rate (e.g. 1200 baud). The data bits contain ASCII text characters that make up the NMEA sentences. This data is directly sampled into the DAS at the DAS sampling frequency and captured in the *.chn file. This function takes in the digitized data and converts back to human readable ASCII:

1. Calculate how many samples should occur for each ASCII symbol. For example, at 1200 baud, this is (5000 sps / 1200 symbols per second) = 4.167 samples per ASCII symbol.
2. Use the gpsFindEdge function to detect the edge sample # of an NMEA sentence burst in the channel data.
3. Look at the middle value of each 4.167 sample interval to determine whether the symbol value is a 1 or 0.
4. Decode this train of 1’s and 0’s into ASCII data. Example:
    a. 10 bits make up 1 start bit, 8 data bits, and 1 stop bit.
    b. If the 8 data bits are 0b00100100 = 36 decimal = ASCII dollar sign character “$”.

5. Look at each consecutive interval until the first null character is found (all 0's), indicating the end of the NMEA sentence.

Main Location and Time Extraction Functions

```
%//$Revision$
%//Copyright © 2017 Diversified Technical Systems, Inc.
%//All Rights Reserved.
%//Function gpsMain
%//Inputs:
%// ppsFile: string of the 1PPS .chn data file name
%//  nmeaFile: string of the NMEA .chn data  file name
%//  baudRate: numerical baud rate of the NMEA data
%//  command: either 'extract' to run a complete extraction or 'summary'
%//        to generate a file summary
%//  outDir: string name of the directory to dump the output results to.
%//Description:
%// This is the main function for the MATLAB GPS extraction utility.
%// Call this function to generate either a summary of extraction data set.

%//$Revision$
%//Copyright © 2017 Diversified Technical Systems, Inc.
%//All Rights Reserved.
%//Function gpsErrorCheck
%//Inputs:
%//  nmeaFile: the filename of the .chn file that stores NMEA data
%//  ppsFile: the filename of the .chn file that stores 1PPS data
%//  baudRate: numerical baud rate value
%//Returns:
%//  returnError: a text error description or a PASS
%//  outfile: string used to form the name of any output files
%//Description:
%//  looks for the following errors in the input files
```

```
%//  -- baud rate is a valid value (1200, 2400, 4800, 9600, 19200, or 38400
%//  -- 1pps and nmea files exist and contain the correct magic number
%//  -- the nmea baud rate and sample rate are compatible
%//  -- that the nmea and 1pps file names are the same indicating they came
%//     from the same dataset.
%//  -- that the files are at least 2 seconds long.
%//  -- that the start and stop percentages make sense
%//  -- that the amount of data is the same in both files
%//  -- that the sample rates are the same in both files


%//$Revision$
%//Copyright © 2017 Diversified Technical Systems, Inc.
%//All Rights Reserved.
%//Function gpsGenerateSummary
%//Inputs:
%//  ppsFilename: name of the DTS .chn file that contains the 1PPS edges
%//  nmeaFilename: name of the DTS .chn file that contains the NMEA
%//  baudRate: the baud rate of the NMEA serial port.
%//  outfile: the name of the output file to store status in
%//Returns:
%//  strSummary:  A text based summary report.
%//Description:
%// This function looks at the beginning and end of the 1PPS and NMEA files
%// and finds the first and last valid 1PPS edges and/or NMEA times. It then
%// generates a summary report of what the files contain.  Only the first
%// and last parts of the files are examined to limit the total amount of
%// data handled during parsing of very large files.
%// The summary includes:
%//   total number of samples
%//   sample rate setting
```

```
%//   estimated total test duration based on the samples and sample rate
%//   Sample that first and last 1PPS rising edge is found
%//   First and last Lat, Long, Time based on GGA or RMC

%//$Revision$
%//Copyright © 2017 Diversified Technical Systems, Inc.
%//All Rights Reserved.
%//Function gpsExtractData
%//Inputs:
%//  ppsFilename: name of the DTS .chn file that contains the 1PPS edges
%//  nmeaFilename: name of the DTS .chn file that contains the NMEA
%//  baudRate: the baud rate of the NMEA serial port.
%//  outfile: the name of the output file to store status in
%//Returns:
%//  strSummary: a text based extract summary report
%//Description:
%// This function first pulls available NMEA and 1PPS data into memory. If both
%// NMEA and 1PPS data are available, it scans through the file, finds 1PPS edges
%// and associated valid NMEA sentences, then dumps 1PPS time and location vs which
%// sample the 1PPS edge fell on to a file.  It also dumps NMEA data to a file.
%// If only NMEA is available, it creates an estimated time output file based on
%// the first rising edge of each NMEA dump.  It also decodes and dumps any NMEA
%// sentences to a file.  Finally, if only a 1PPS file is available, it dumps
%// the sample number that each 1PPS rising edge occurs at to a file.  The script
%// generates a summary of the extracted data and returns this to the main function
%// for status reporting.  If there are fewer valid NMEA or 1PPS reports than there
%// should be given the length of the record in seconds, a warning about possible
%// corrupted data is created.
```

Subfunctions

```
%//$Revision$
```

```
%//Copyright © 2017 Diversified Technical Systems, Inc.
%//All Rights Reserved.
%//Function gpsGetChnData
%//Inputs:
%//  fid: File ID handle
%//  startSample: The beginning index of the data to return
%//  endSample: The ending index of the data to return
%//Returns:
%//  data:  An array of samples pulled from the input file
%//  sampleRate: the numerical sample rate stored in the DTS file header
%//  nSamples: the total number of samples as indicated in the header
%//Description:
%//  This function opens up the DTS .chn data, pulls the sample rate and
%//  number of samples out of the header data and optionally returns
%//  samples from the datafile.  The startSample and endSample values
%//  allow the user to pull a subset of the total data.  The data is
%//  thresholded and returned as either 0's or 1's. The thresholding
%//  algorithm looks at the maximum values and minimum values in the data,
%//  sets the threshold at a mid-point between the two adjusted up from
%//  the minimum value.  To account for the inversion of the 1PPS and NMEA
%//  into the DAS differential amplifier (-SIG), it then sets any sample
%//  higher than this threshold to 0 and sets any sample equal to or lower
%//  than this value to a 1.  If the threshold is significantly different
%//  from an ideal value, the ideal value is used. This may occur if the
%//  cable was damaged in the early part of the captured data or some other
%//  unforeseen exception.

%//$Revision$
%//Copyright © 2017 Diversified Technical Systems, Inc.
%//All Rights Reserved.
%//Function gpsFindEdge
```

```
%//Inputs:
%//  sampleTrain: array of sampled ADC data that contains a 1PPS edge or NMEA
%//              data transitions
%//  Fs: the sample frequency that the data is taken at
%//  baudRate: numerical baud rate value of the NMEA data only.
%//Returns:
%//  edgeLocations: The sample numbers at which the edges are found in the
%//                data
%//Description:
%// This function first filters the ADC data into 1 or 0 values.  It then
%// slides a boxcar along the sample train looking for a period of zeros
%// followed by a 1 which indicates a rising edge at the start of either a
%// 1PPS signal or NMEA sentence.  The boxcar width is set to be a little
%// longer than the number of sample's found in a single ASCII character at
%// the specified baud rate and sample rate.  This allows the algorithm to
%// discern 0's for both the NMEA and 1PPS data.

%//$Revision$
%//Copyright © 2017 Diversified Technical Systems, Inc.
%//All Rights Reserved.
%//Function gpsDecodeNmea
%//Inputs:
%//  sampleTrain: array of sampled ADC data that contains a 1PPS edge or NMEA
%//              data transitions
%//  Fs: the sample frequency that the data is taken at
%//  baudRate: numerical baud rate value of the NMEA data only.
%//Returns:
%//  strNmea: a string of all NMEA sentences within the sampleTrain.
%//Description:
%//  This function scans through the sampleTrain which contains NMEA serial
%//  data at the specified baudrate @ 8n1.  It then decodes serial data
```

```
%//  into ASCII character values (if these are present) and appends them
%//  to the return string.


%//$Revision$
%//Copyright © 2017 Diversified Technical Systems, Inc.
%//All Rights Reserved.
%//Function gpsParseTimeLoc
%//Inputs:
%//  nmeaString: an ascii string of NMEA sentences with CSV values
%//Returns:
%//  valid: a 1 indicates a valid RMC or GGA was found.  0 otherwise
%//  hr, min, sec:  a time string in this format 'HH', 'MM', 'SS.SSSS'
%//  latD, latM, latH: latitude string of degrees, minutes, or hemisphere
%//                  (N or S) in this format 'DDD', 'MM.MMMM', 'H',
%//  longD, longM, longH: longitude string of degrees, minutes, or hemisphere
%//                  (E or W) in this format 'DD', 'MM.MMMM', 'H',
%//  day, month, yr: date in this format 'DD', 'MM', 'YY'
%//Description:
%// This function splits the nmea sentences into comma separated values
%// it then looks for the first gga or rmc message and then pulls the
%// time, latitude, latitude hemisphere, longitude, longitude hemisphere
%// string out of the sentence and returns these strings.
```

Revision History

Rev	Date	By	Description
0.5	13 Mar 2018	AM	Updated remaining photos
0.4	22 Feb 2018	AM	Updated to reflect changes noted during training.
0.3	15 Feb 2018	DC	Adjusted spacing on page 1. A few edits for readability.
0.2	15 Feb 2018	AM	Redlines for readability, updated function headers for accuracy with adjusted code, updated figures for accuracy of sensor configuration and configuration instructions
0.1	31 Jan 2018	AM	Redlines for readability
0	3 Jan 2018	DC	Document Created