REV 1.3

GPS Workbench Data Formats Descriptions

iTrax02 Evaluation Kit

1.08

A short description of the document content A short description of the document content A short description of the document content A short description of the document content A short description of the document content A short description of the document content A short description of the document content A short description of the document content

December 9, 2002

Fastrax Oy

www.fastrax.fi

FastraX

CHANGE LOG

Rev.	Notes	Date
1.0	Initial Revision	14-03-2002
1.1	Layout changes and corrections to figures 1 and 2.	09-12-2002
1.2	Updated rev number	10-12-2002
1.3	Updated page numbers	11-12-2002

CONTENTS

COMPLEMENTARY READING

The following reference documents are complementary reading for this document:

Ref. #	File	Document
01	Install.pdf	iTrax02 Evaluation Kit: Installation Guide
02	UGuide.pdf	iTrax02 Evaluation Kit: GPS Workbench Users Guide
03	iTalk.pdf	iTrax02 Evaluation Kit: iTalk Protocol Specification

1. GENERAL

This document describes the iTalk data formats. Structure definitions in iTalk protocol specification (Ref: [03]) define not only iTalk messages but also iTalk data formats, which are essentially the same. Actual structures are not reprinted here.

Due to current software development phase all described data formats are subject to change without notice.

1.1 Data types

Data type names used in this document refer to portable data types as follows:

Integers:

- BYTE is 8bit unsigned integer
- WORD is 16bit unsigned integer
- DWORD is 32bit unsigned integer
- INT16 is 16bit signed integer
- INT32 is 32bit signed integer
- SHORT is same as INT16
- LONG is same as INT32

Float:

- DOUBLE is 64bit IEEE-double precision float

1.2 Float value conversions

Floating-point numbers are stored as standard 4 word long IEEE-doubles in data files. iTrax02 however uses custom 3 word long floating point numbers. iTalk protocol specification (Ref: [03]) describes how to convert these values from and to standard 4 word long IEEE-double values.

Note! 48-bit VS_DSP double values mentioned in message structures of iTalk protocol specification (Ref: [03]) are stored as 64-bit IEEE-doubles in data files.

2. FILE FORMATS

Each **iTalk message structure** defined in iTalk protocol specification (Ref: [08]), (i.e., EPHEMERIS_MSG) corresponds to one **data name** (i.e., named EPHEMERIS). If the Fastrax GPS Workbench archiving function is activated, it stores user selected message structures to their corresponding files. No two different message structures are stored under the same file name. The following tables list the achievable message structures and its corresponding file-id and filename, under which these message structures are stored.

Data name	iTalk Message structure	File id	Default file name where the message structure are stored
PSEUDO_DATA	PSEUDO_DATA_MSG	1	PseudoData.dat
UTC_IONO	UTC_IONO_MSG	4	UtcIono.dat
TRACK	TRACK_MSG	5	Track.dat
ACQ	ACQ_MSG	6	Acq.dat
NAVIGATION	NAVIGATION_MSG	7	NavLSQ.dat
NAV_KALMAN	NAV_KALMAN_MSG	8	NavKalman.dat
PRN_STATUS	PRN_STATUS_MSG	9	PrnStatus.dat
CUSTOM_FIX	CUSTOM_FIX_MSG	10	CustomFix.dat
AIDING	AIDING_MSG	11	Aiding.dat
SUBFRAME	SUBFRAME_MSG	12	Subframe.dat
AGC_CONTROL	AGC_CONTROL_MSG	13	Gain.dat
ACQ_AIDING	ACQ_AIDING_MSG	(14)	Not archived. File id reserved for future use.
EPHEMERIS	EPHEMERIS_MSG	15	Ephemeris.dat
NAV_PARAMS	NAV_PARAMS_MSG	41	NavParams.dat
KALMAN_PARAMS	KALMAN_PARAMS_MSG	(42)	Not archived. File id reserved for future use.
ADV_PARAMS	ADV_PARAMS_MSG	(43)	Not archived. File id reserved for future use.
NAV_START	NAV_START_MSG	51	NavStart.dat
NAV_STOP	NAV_STOP_MSG	52	NavStop.dat
NAV_STATE	NAV_STATE_MSG	(53)	Not archived. File id reserved for future use.
NAV_START	NAV_START_RECONFIG_MSG	(54)	Not archived. File id reserved for future use.

Figure 1 iTalk message structures and its corresponding archive files. See Ref: [03] for iTalk message structure definitions.

Simulated Data name	iTalk Message structure	File id	Default file name where the smessage structure are stored
NAVIGATION	NAVIGATION_MSG	107	NavLSQ(Simulated).dat
NAV_KALMAN	NAV_KALMAN_MSG	108	NavKalman(Simulated).dat
CUSTOM_FIX	CUSTOM_FIX_MSG	110	CustomFix(Simulated).dat
EPHEMERIS	EPHEMERIS_MSG	115	Ephemeris(Simulated).dat
PSEUDO_DATA	PSEUDO_DATA_MSG	101	PseudoData(Simulated).dat
PRN_STATUS	PRN_STATUS_MSG	109	PrnStatus(Simulated).dat

Figure 2 Simulated iTalk message structures and its corresponding archive files. See Ref: [03] for structure definitions.

iTalk message structures are written sequentially into its corresponding archive file. i.e., NavParams.dat consists of continuous array of NAV_PARAMS structures.

2.1 File numbers

When FastraX GPS Workbench archives messages into files, it names the files according to **registry keys**. All these keys and their values can be found under the key "*Fastrax Oy\Fastrax GPS Workbench*".

Note! Usually there is no need to change filenames. Readers are suggested to skip the rest of this section.

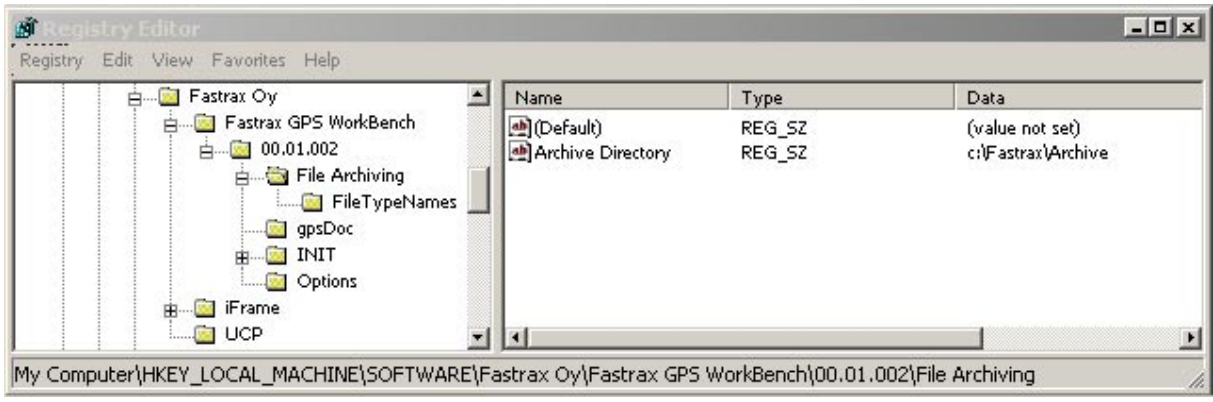

Figure 3 Archive directory setting in registry

Archive directory is defined by registry key "*File Archiving"* value "*Archive Directory"*. As an example: "C:\archiving".

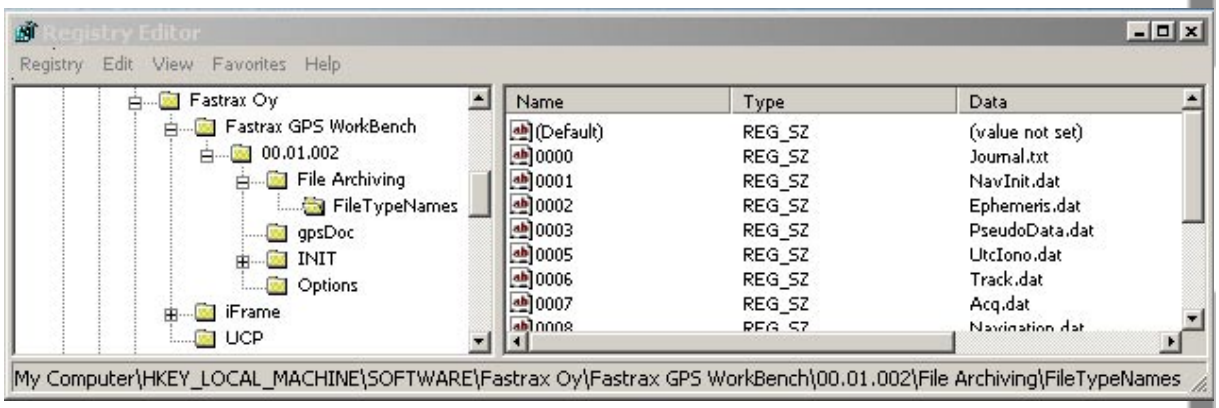

Figure 4 Default filenames in registry

Individual archive filenames are defined by sub-key of "*File Archiving"* named "*FileTypeNames"*. Value names refer to file id and value data to actual filename.

2.2 Order of messages

In addition to the archive datafiles, GPS Workbench also creates two other files named *Archive.ini* and *Journal.txt.* (Latter can be named otherwise in registry.)

Archive.ini lists the names of individual archive files as they are read from the registry. This arrangement allows user to copy archive files from a computer to another even if filenames in registry differ between these computers. When GPS Workbench reads archived data, it tries to open the files by their names defined here.

Journal.txt (or journal file) lists the messages with time stamps. It lists all the messages with time stamped at the moment they were received and thus makes it possible to pick certain desired messages from certain files in certain time when running archived data.

2.3 Archive.ini

As mentioned above Archive.ini lists filenames which were used when archiving for each file id. It is a text file with structure shown as follows:

line	text value	description
1	[Files]	Header string
2	0000=<*journal_file*>	…where <*journal_file*> is the name of the journal file.

line	text value	description
3-n	000*n*=<*file_name*>	…where n is the file id marked in 4 digits format and begins with zero-headings and <*file_name*> is the corresponding file name.

Figure 5 Archive.ini

```
[Files]
0000=Journal.txt
0001=NavParams.dat
0007=Acq.dat
0006=Track.dat
0002=Ephemeris.dat
0003=PseudoData.dat
0009=PrnStatus.dat
0008=Navigation.dat
0010=CustomFix.dat
0005=UtcIono.dat
0108=Navigation(Simulated).dat
0110=CustomFix(Simulated).dat
```

Figure 6 Example of Archive.ini

Note! This file description is for information only. Do not attempt to edit Archive.ini file.

2.4 Journal file

Journal file lists the time-stamped messages. It is a text file with structure shown as follows:

Line	text value	description
1	[JOURNAL]	Header string
2	0=0,0,<*x*>,<*init_msg*>	…where <*init_msg*> is arbitrary version/copyright message.
3-n	<*n-2*>=<*ms*>,<*msg_id*>,<*length*>	…where n is the message number (being line number – 2), <*ms*> is the millisecond timestamp, <*msg_id*> is id of the file where this message should be read from, <*length*> describes how long the message is (How many

Line	text value	description
		words should be read from appropriate .dat file. Measured in BYTEs.

Figure 7 Data structure in Archive.ini

```
[JOURNAL]
0=0,0,54,(C) 1999-2000 FASTRAX file archive v1.1 - Journal file
1=170,1,56
2=781,7,74
3=3685,7,74
4=3976,6,294
5=4206,6,294
6=4456,6,294
7=4707,6,294
8=4967,6,294
9=5208,6,294
…
```

Figure 8 Example of the journal file

i.e. Provided that the example journal file above be read and the first message be sent after 170ms of execution, “PseudoData.dat” file would contain 56 BYTEs long PSEUDO_DATA structures.

If 781ms has elapsed since the start of the execution, 74 BYTEs long NAVIGATION structure (from NacLSQ.dat file) would be sent in the corresponding message and so on…

Note! This file description is for information only. Do not attempt to edit the journal file