

AMENDMENT LIST NUMBER 4 TO
TECHNICAL SPECIFICATION
FOR IGC-APPROVED GNSS FLIGHT RECORDERS

ISSUED 25 MAY 2001 BY FAI
ON BEHALF OF THE INTERNATIONAL GLIDING COMMISSION

1. Glossary. Add the following:

API - Application Programming Interface. A set of functions that an application can call to tell the operating system to perform a task. (AL4)

DLL - Dynamic-Link Library. In Microsoft Windows, a DLL is a small program containing functions that other programs or resources can call or use. Outside MS Windows, DLLs are used in areas such as Distributed Interactive Simulation (DIS) links and other processing. (AL4)

DSA - Digital Signature Algorithm. In its specialist meaning, an asymmetric system of Public/Private Key Cryptography (PKC) used in the US National Institute of Standards and Technology Digital Signature Standard (DSS). It is comparable in performance and strength to an RSA (qv) signature with the same key length, and uses a protocol called SHA-1 as the message digest algorithm. Signing a message takes about 1/2 the computation of RSA thus reducing data transfer times from FR to PC, and some computation can be done "on the fly" while the recorder is operating normally. However, DSA takes more computation than RSA to verify a signature, the IGC VALI process taking longer than RSA (but the VALI process is not time-critical, whereas data transfer from FR to PC is). More detail on the implementation of DSA can be found via <http://csrc.nist.gov/publications/fips/fips186-2/fips186-2.pdf>. GFAC will give advice as necessary. (AL4)

Horizontal fix accuracy - the best prediction for the horizontal 2-sigma error of the overall position error. Included in the IGC data file in the B (fix) record through the FXA three-letter code.

NMEA - add: "NMEA data is divided into groups called "sentences" identified by three-letter codes, the details being given in documents such as NMEA 0813. For instance the sentence GGA gives GPS fix data, GNS gives fix data for all GNSS systems (US GPS, Russian GLONASS, European Galileo and any other systems), GSA gives the satellites in view at any one time. Some GNSS receiver boards output NMEA data directly and others use manufacturer=s binary or other output formats. In the latter case, where NMEA data is mentioned in this document the FR manufacturer must show that equivalent data that is acceptable to GFAC is recorded on the IGC data file."

PGP - Pretty Good Privacy. A commercial system for electronic security that uses RSA asymmetric keys, first publicised by Philip R Zimmerman in June 1991 through a public Internet bulletin board. The US authorities initially tried to prosecute Zimmerman for a security breach, but after 3 years gave up the attempt. See <http://www.pgp.com>. The rights of Zimmerman=s company PGP, Inc., were later sold to Network Associates, <http://www.nai.com>. It has been estimated that some over 500 million copies of PGP are in use worldwide, and the padlock= symbol on a PC screen normally indicates that the PGP system is available. (AL4)

PKC - Public/private Key Cryptography. A system where the recipient of a message has an encryption system that is not secret (the Public Key) and is used by people sending messages to him. However, the mathematical factors that make up the Public Key are only held by the recipient (the Private Key), and are needed before the message can be de-coded. The PKC principle was discovered in May 1975 by Whitfield Diffie, Martin Helman and Ralph Merkle (DHM) of the Electrical Engineering Department of Stanford University, USA, and previously in 1973 by James Ellis and Clifford Cocks of the classified Government

Communications HQ organisation in the UK. The first commonly available practical application of PKC was the RSA system (qv). (AL4)

Amend Pressure Altitude to the following (first sentence not changed, rest allows use of instrument static (IGC decision):

Pressure Altitude - In a GNSS FR, this is a five numeric group indicating the pressure altitude in metres with respect the International Standard Atmosphere (ISA) used in aviation, to a sea level datum of 1013.25 HPa. The pressure recorded in the *.IGC file may either be "cockpit static" (vented within the FR box), or use a tube connection to the pressure from glider instrument system static tubing. If the pressure altitude signal within the FR is used for other purposes such as cockpit instrument readings which can be set to other datums such as QNH or QFE, a one-way transmission system must be used from the sensor so that the IGC file always records the required ISA to the 1013 sea level datum irrespective of other settings used for flight instruments. The permitted use of instrument-static is intended for a GNSS FR mounted in the instrument panel. With such an installation, an OO as part of the inspection of the FR installation must check the tubing and the pressure connection to the FR to ensure that they will be out-of-reach of the aircrew in flight. This is to prevent alteration to the IGC-file pressure altitude record by any method. (AL4)

RSA - A system of Public/Private Key Cryptography (PKC), developed by Ronald Rivest, Adi Shamir and Leonard Adelman of MIT, employing an asymmetric system for key exchange. First published in an article in Scientific American in August 1977, and the company RSA Security Inc was formed to apply it commercially. More detail on the implementation of RSA can be found in the book "Applied Cryptography" by Bruce Schneier, 2nd edition, ISBN 0-471-11709-9. An overview of various cryptographic algorithms can be found in <http://www.ssh.fi/tech/crypto/algorithms.html>. High Speed RSA Implementation (PDF file) is in: <ftp://ftp.rsasecurity.com/pub/pdfs/tr201.pdf>. Details of the FIPS 180 Secure Hash Standard are in <http://www.itl.nist.gov/fipspubs/fip180-1.htm>. Cryptographic libraries with source code in C and C++ are in: <http://www.cs.auckland.ac.nz/~pgut001/cryptlib> and <http://www.eskimo.com/~weidai/cryptlib.html>. GFAC will give advice as necessary. (AL4)

Vertical fix accuracy - the best prediction for the vertical 2-sigma error of the overall position error. When included in the IGC data file, through the VXA three-letter code.

2. GFAC member=s period of office. 1.2.1 to read: GFAC members will be appointed for a period agreed by IGC ... (IGC decision March 2001)

3. Sending information and hardware.

1.3 Middle sentence to read: The appropriate fee shall be deposited in FAI's IGC account by each applicant when hardware is first sent for evaluation (initially only to the GFAC Chairman, see 1.3.3.1 below). IGC approval will not be given until the appropriate fee is received and all expenses attributable to the manufacturer have been paid to FAI.

1.3.3.1 to read: For the submission of any new model of FR, first send details to the GFAC Chairman of the intended design such as draft specifications, drawings, draft manual, commonality with existing models, etc., as soon as a draft is available. The Chairman will send such details to GFAC members and appropriate technical experts and will co-ordinate comments that will be sent back to the manufacturer. Email is the recommended method in the form of text or attached files in word-processed format (such as MS Word). Use jpg (compressed) format for diagrams and pictures at not more than 200kB per graphic unless requested otherwise. As soon as IGC-format files are available from early hardware, send copies to the GFAC chairman so that the exact format can be checked and commented on. Do not send any hardware until GFAC comments have been made on the written Specification and initial IGC files have been produced and sent, unless advised otherwise by the GFAC Chairman. In terms of sending hardware, as soon as a complete prototype or alpha/beta test version is available, send a single example to the GFAC Chairman for initial evaluation and feedback, before the fix-of-design stage is reached. The Chairman=s

evaluation team will test the hardware and report to GFAC members, relevant technical experts and the FR Manufacturer. When hardware is sent, the FR manufacturer should apply to FAI for IGC-approval and pay the appropriate fee. All GFAC members have a right to ask for hardware for testing themselves. Therefore, after appropriate correspondence between the Chairman and the FR manufacturer, and after any necessary changes have been made to the prototype equipment evaluated, sets of hardware shall then be sent by the manufacturer to those GFAC members notified to the manufacturer by the Chairman. (AL4)

1.3.3.1.1 Fix Error Rates. Recorder systems must show an error rate of fixes better than 1 in 1000 fixes in normal signal propagation conditions. Error rates are taken from the centre of fixes assessed as in excess of 100 metres from a true 2-D (lat/long) position (discounting any errors in GNSS altitude). General accuracy must be to an average of better than 20 metres in lat/long (assuming that the Selective Availability or equivalent error is not applied by the GNSS control authority). (AL4)

1.3.3.1.2 Pressure Altitude Calibration. The pressure altitude recording system in the FR must be calibrated using standard procedures for barograph calibration, and a calibration table and the IGC file for the calibration forwarded with any hardware that is sent.

4. Evaluation period, para 1.4. Last sentences to read: The evaluation period starts when all members of GFAC who have expressed a wish to test the hardware themselves, have received all of the required equipment and documentation in good order and ready to test. The GFAC Chairman will notify the manufacturer of the contact details of individuals to whom hardware should be sent (see 1.3.3.1). If such individuals are not provided with all of the equipment to be evaluated, the target evaluation period does not apply although the evaluation will be completed as soon as practicable in the circumstances prevailing. This will involve equipment being sent from individual to individual. Any excess expenses incurred by individuals (such as postal, excise and tax), shall be paid by the FR manufacturer into the FAI account on request so that individuals can be re-imbursed and do not have to pay these expenses themselves. (AL4)

5. Submission dates. Add to 2.1.3: The date of submission is taken as the date of receipt of the appropriate fee by FAI or the date of receipt of all equipment required for testing by the individuals notified to the manufacturer by the GFAC Chairman who are to receive the hardware (see 1.3.3.1), whichever date is the later. (AL4)

6. Type of GNSS receivers. Para 2.4, add "a receiver capable of processing data from at least 12 satellites at one time".

7. Connectors on the outside of the Recorder case.

Para 2.7.2.2.2, add "Where the antenna connector on the FR case is not the 9mm diameter BNC bayonet type, for GFAC testing a short interface cable must be provided so that antennas with BNC connectors can be used for test purposes. Since antennas with BNC connectors are commonly available, in the case of FRs not having a BNC connector on the FR case, it is recommended that production equipment includes an interface cable so that a BNC antenna connection can be used if the original antenna has to be replaced (such as during a competition)."

Para 2.7.2.2.7, last words to read: one of which shall be available on the outer case of stand-alone FRs. In the cases of panel-mounted FRs an interface connector may be used so that one of the IGC-approved connectors is readily available (such as on the end of a short cable connector from the FR itself) at the back of the instrument panel (or in another convenient place) for data transfer purposes.

8. Electronic Security.

8.1 Para 2.8 on security principles, add: Individual FRs must have different security keys to others, so that if the key for one FR is broken, the rest of the product range will still be secure. (AL4)

8.2 Para 2.8.3.1 on Message Digest, to end as follows:

In the case of RSA, a key of at least 512 bits will be required, and, for non-RSA systems, a key length giving equivalent security as decided by GFAC. With progress in computing and the possibility of even asymmetric cryptographic systems being broken, such key lengths and other security aspects will be kept under review and revised from time to time. It is expected that new designs of GNSS FRs will have more capable processors and more memory than earlier designs, so that larger key sizes should not present a problem. The DSA system is a permissible alternative to RSA and may give shorter times for data transfer from the GNSS FR to a PC. With the DSA system it may be possible to make security calculations while the FR is recording data, which may save time during data transfer to a PC after the flight. A 100kB IGC data file should transfer with full security encoding from the FR to a typical PC used by gliding competition organisers (typically a low range Pentium, but this level will be kept under review), in about 1 minute or less. The time to execute the VALI-XX.EXE program is considered to be less critical. For more information on RSA and DSA, see the Glossary, and for advice on these matters, consult GFAC, initially through its Chairman. (AL4)

9. DATA-XXX program file. 2.9.3.1 third sentence to read "The data from the last flight must automatically be transferred to the PC to the same directory as the DATA program file, without needing special switching or keyboard actions "

10. Use of Palm PCs, PC-cards etc. Add at end of 2.9.3.3: It must be possible to transfer data from the secure storage medium of the FR directly to the serial port of a low-specification PC (such as at the glider on the field, using a 486/33 laptop PC) and to create the IGC file for the flight on the PC hard disk or a standard 1.4MB floppy disk (which may be self-booting), by use of the DATA-XXX.EXE program. For this data transfer, the connector fitted to the body of the FR storage device must be one of the options specified in para 2.7.2.2.7, or, for FRs that are permanently mounted in an instrument panel, through a short interface cable ending in one of the IGC-approved connector options. This is so that pilots, OOs and competition organisers only have to provide facilities for the connectors specified rather than a larger number of other connectors. In addition to this basic method of transferring flight data from the FR storage to a PC, other methods of creating and storing the IGC file are permissible as long as the resulting IGC file is identical to that produced by the basic method, and satisfies any later checks made against the VALI-XXX.EXE program. Other methods include the use of separate software programs which include data transfer (such as part of a manufacturer's integrated program for setup, transfer and analysis), also the use of PalmPCs and PC cards for data transfer from the FR storage so that storage of secure IGC files is ensured before the analysis and validation process. If in doubt about a particular type of device, consult GFAC beforehand. Such methods and devices will be tested by GFAC during the evaluation of the FR and any storage devices to ensure that the IGC data file produced is identical to that produced after the basic process using the DATA-XXX program file on a low-range PC.

11. Windows programs. 2.9.3.6.1 - Amend first sentence to read: "FR Manufacturers may also provide direct Windows equivalents to the DATA, CONV and VALI DOS program files, following the protocols given in Appendix 3." Last sentence, add: "Also see below on long file names", and delete the existing note after the last sentence.

12. Manufacturer=s Codes. Appendix 1 para 2.5.7 add: W - Westerboer - WES

13. Valid fix flag. Change Appendix 1 para 4.1 to: Fix valid 1 byte. Use A in the IGC file to denote a 3-D fix and V for a 2-D fix. Where data in NMEA format is used within the FR, in the GSA sentence (DOP and active satellites), put A in the IGC file for GSA mode 3, and V for GSA mode 2. In the future, other characters may be used in this field in the B record to denote other meanings, and will be notified in future amendments. (AL4)

Also in 4.1, add to B record format: All of the information within each B-record must have a data issue time within 0.1 seconds of the time given in the B-record. Where NMEA data is used within the FR, fix data should be taken either from the GGA or GNS sentences. GGA is specific to the US GPS system. GNS

is intended for all GNSS systems (GPS, GLONASS, Galileo and future systems), and should be used if it is available from the GNSS board installed. The three characters for FXA and two for satellites in use (SIU), should be derived from parts of the fix sentence used and are described in both the NMEA GGA and GNS as `Horizontal DOP=` and `number of satellites in use=`. In the B Record FXA (HDOP) should be recorded as a three-figure group in metres and SIU as a two group number. SIU is an optional record and may be used to back up the more detailed satellite data in the mandatory F-record. Leading zeros should be included as necessary. Because earlier IGC-approved GNSS FRs may not have FXA and SIU in their B-records, the position of this data in each B record line must be indicated (for instance to analysis programs) by including them in the I record which designates the positions of additional fields in the B record. FXA should be placed after the two groups for altitude, followed optional fields such as SIU and then ENL for Motor Gliders. FXA would therefore normally occupy bytes 36, 37 and 38, and SIU bytes 39 and 40 in each B-record line. (AL4)

14. Constellation (F-record). Appendix 1 para 4.3, change the narrative to read: This is a mandatory record. For the US GPS system, the satellite ID for each satellite is the PRN of the satellite in question, for other satellite systems the ID will be assigned by GFAC as the need arises. Where NMEA data is used within the FR, the ID should be taken from the GSA sentence that lists the IDs of those satellites used in the fixes which are recorded in the B record. The FR Record is not recorded continuously but at the start of fixing and then only when a change in satellites used is detected. (AL4)

15. Three letter codes. Appendix 1 para 7.

REX. After `Manufacturer defined data=` in REX entry, add `defined in the I or J record as appropriate, normally in the form of a TLC (which, if a new variable is agreed, may be a new TLC allocated by GFAC at the time). (AL4)

New entry: SIU - Satellites in use. A two-character field from the NMEA GGA or GNS sentences, as appropriate, or equivalent data agreed by GFAC. (AL4)

VXA. Vertical fix accuracy. Add: Three characters in metres from the VDOP part of the NMEA GSA sentence, or equivalent data agreed by GFAC.

16. Appendix 2 - Sample Test Schedule. Add new para on MoP testing:

8.5 Tests will be made on any system for recording the operation of the Means of Propulsion (MoP) for motor gliders. Where Engine Noise Level (ENL) or vibration sensors are used, tests will be made with the FR in a number of types of glider and motor glider. These will include glass-construction machines with low aerodynamic cockpit noise, also machines with higher cockpit noise in gliding flight. Conditions will include speed and other variations, and sideslipping flight with cockpit panel(s) open (producing the so-called `organ-pipe=` effect in some gliders). Operation of both two-stroke and four-stroke engines will be tested at all available power settings. Results will be analysed to ensure that a clear difference in the IGC file data is shown between all types of gliding flight, and any engine running at positive thrust settings. If these conditions are not shown, modifications to the ENL system will be required until they are.

17. Windows programs -new appendix 3:

Appendix 3 to
Technical Specification for
IGC-approved GNSS Flight Recorders (AL4)

Windows-based short programs for data transfer and conversion to IGC format, and validation of data

1. INTRODUCTION.

1.1 General. This appendix describes an IGC standard for Manufacturer-supplied Windows Dynamic Link Libraries (DLLs) in a 32-bit Windows environment, for the DATA, CONV and VALI functions that are described in para 2.9 of the main part of this Specification.

1.2 Availability on the IGC GNSS web site. The DLLs shall be freeware and be made available through links from the IGC/GNSS web site <http://www.fai.org/gliding/gnss>. IGC will supply a sample control program, in both source and executable form. This will load a DLL and call each of the Application Programming Interface (API) functions specified below. This control program will also be available through links from the IGC web site.

1.3 Functions supported. All such DLLs shall support the functions described in the API below.

1.4 Control program. The Control Program will check responses from the DLLs and also perform checks for the existence of a file before calling on a DLL to open it for reading (e.g. ConvertLog or ValidateLog) and to query overwriting an existing file (e.g. DownloadLog and ConvertLog). The Control Program will also select the COM port to be used.

1.5 DLL Naming. The name shall be of the form IGC-XXXy.DLL where XXX is the Manufacturer's three Letter Code as defined in Appendix 1 para 2.5.7. It is expected that a manufacturer's basic DLL will be able handle all GNS FRs in the product range. In case this cannot be achieved, the symbol "y" above is an optional alphanumeric for other DLLs from a given manufacturer.

2. APPLICATION PROGRAMMING INTERFACE (API) FOR MANUFACTURER-SUPPLIED DLLs

2.1 General. A standard API is described below in para 3. It includes the functions mentioned in para 1.1 for the operating systems Windows 95/98/ME, NT/2000, and subsequent releases. The API shall be implemented by a DLL supplied by each flight recorder manufacturer, which exports a defined set of functions for use by control programs. These control programs may include third-party flight evaluation applications, competition scoring software, and minimal generic programs for the use of pilots, official observers and contest directors.

2.2 Control programs. The DLLs will be stored in a common directory on the computer of the end-user. A control program will use the Win32 *LoadLibrary* or *LoadLibraryEx* functions to load a DLL using run-time linking. The control program will then query it for the entry point of each API function by name, using the Win32 *GetProcAddress* function.

2.3 Functions and descriptions. Some of the functions described below are designated optional. If a function is not implemented by a particular DLL, *GetProcAddress* will return NULL. In the API descriptions below, DWORD, BOOL, TCHAR, LPTSTR, and LPCTSTR are standard Win32 API types defined as long, bool, char, char *, and const char *, respectively, for the required ANSI (as opposed to UNICODE) DLL build. HWND is a 32 bit window handle. FALSE is integer 0, TRUE is any non-zero integer value.

2.3.1 Icons. In addition to these functions, each DLL is to provide a unique 32x32x4 (16 color) icon, with the fixed resource identifier IDI_IGCICON (defined to be 101). Control programs may use the Win32 *LoadImage* call to obtain the icon, and display it in a listbox or other context.

3 **API SUB-ROUTINE DESCRIPTIONS**. The standard API follows. Titles of main sub-routines are in bold, underlined, and end in the letters DLL, FR or LOG. They are followed by relevant data such as parameters, return values, and remarks. Paragraph numbering is not used, and the sub-routine main title and sub-para title should be used when referring to a particular item.

IdentifyDLL

```
DWORD IdentifyDLL(LPTSTR value, DWORD size)
```

The *IdentifyDLL* function obtains an identifying string, which the control program will enter in a listbox used to select the appropriate DLL.

Parameters

value
[out] pointer to buffer that will receive string.

size
[in] size of the buffer pointed to by *value*.

Return Values

Function returns number of bytes in returned string, if actual length of string exceeds *size*, the string will be truncated to *size* -1 bytes.

Remarks

The string consists of five fields, separated by the "pipe" character ("|", 0x7C), the manufacturer name, supported FR name(s), DLL software revision number, and two comma separated lists of zero or more file extensions. The first list of extensions identifies manufacturer proprietary log files, if any, which can be converted to IGC format using the *ConvertLog* function. The second list identifies log files (possibly including IGC format) that can be authenticated by the *ValidateLog* function. A terminating NUL character is always appended to the string (but not included in the returned count). Maximum permitted length of the string (excluding the terminating NUL character) is 127 characters. Example:

```
Acme Instruments|XL 100, 200|2.0|XL1,XL2|XL1,XL2,IGC
```

InitializeDLL

```
void InitializeDLL(HWND mainWindow)
```

InitializeDLL is an initialization function that must be called before any of the other functions, with the exception of *IdentifyDLL*.

Parameters

mainWindow
Handle for the control programs main window, or NULL if there is none.

Remarks

The window handle should be stored in the DLL, and used as the parent handle for any dialog boxes displayed

by DLL functions. These dialogs should be centered within the main window.

KeepAwakeIntervalDLL

```
DWORD KeepAwakeIntervalDLL()
```

The *KeepAwakeIntervalDLL* function is used to obtain the nominal time interval between calls to *KeepAwakeFR*.

Return Values

Returns the interval in milliseconds. If 0 is returned, *KeepAwakeFR* calls are not required (and will be ignored).

SerialOptionsDLL

```
DWORD SerialOptionsDLL(LPTSTR options, DWORD size)
```

[OPTIONAL] The *SerialOptionsDLL* function displays a modal dialog box requesting any user settable connection options (line speed, flow control, etc.) needed to configure a serial port for use with the manufacturers FRs.

Parameters

options

[out] pointer to buffer which will receive the connection options.

size

[in] size of the buffer pointed to by *options* in bytes.

Return Values

Function returns number of bytes in returned string, if actual length of string exceeds *size*, the string will be truncated to *size* - 1 bytes.

Remarks

This dialog should not include selection of the serial communication device. Maximum permitted length of the returned string (excluding the terminating NUL character) is 63 characters. The string is intended for use in a subsequent call to *SerialConnectFR*, the actual format of the string is determined by the manufacturer. The control program may choose to store this string in the registry or a file for use in future sessions.

This is an optional function, and should only be exported by the DLL if there are user settable connection options.

SerialConnectFR

```
BOOL SerialConnectFR(LPCTSTR device, LPCTSTR options)
```

The *SerialConnectFR* function is used to establish communication with a FR connected through a serial port. Must be called prior to using *KeepAwakeFR*, *IdentifyFR*, *IdentifyLogFR*, *DownloadLogFR*, and/or *DisconnectFR*.

Parameters

device

[in] name of the serial communication device device ("COM1", etc.).

options

[in] string returned by a previous call to *SerialOptionsDLL*, or NULL to use the default device options. The format of this string is determined by the manufacturer.

Return Values

Returns TRUE if connection established, FALSE otherwise.

Remarks

If a connection cannot be established, the function should display a modal dialog box detailing the problem.

KeepAwakeFR

```
BOOL KeepAwakeFR ()
```

The *KeepAwakeFR* function is used to prevent the FR from disconnecting during idle periods between calls to *ConnectFR* and *DisconnectFR*. If *KeepAwakeIntervalDLL* returns a non-zero value, the control program must call *KeepAwakeFR* each time that interval elapses.

Return Values

Returns TRUE if the FR still connected or FALSE if connection has been broken.

IdentifyFR

```
DWORD IdentifyFR(LPTSTR value, DWORD size)
```

The *IdentifyFR* function is used to obtain the manufacturer id/serial number, the FR model name/number, and the FR sealed status for the connected FR.

Parameters

value

[out] pointer to a buffer which will receive the string result.

size

[in] size of the buffer pointed to by *value* in bytes.

Returned Values

Function returns number of bytes in returned string, if actual length of string exceeds *size*, the string will be truncated to *size* -1 bytes.

Remarks

The string consists of three fields, separated by the "pipe" character ("|", 0x7C), the manufacturer id/serial number (formatted MMMNNN, where MMM is the manufacturer id, and NNN is the serial number), the FR model name/number, and the FR sealed status (ASEALED@ if sealed, AUNSEALED@ if not). Maximum permitted length of the string (excluding the terminating NUL character) is 63 characters. Example:

```
AXL01F|XL 100|SEALED
```

IdentifyLogFR

```
DWORD IdentifyLogFR(DWORD index, LPSTR value, DWORD size)
```

The *IdentifyLogFR* function is used to obtain information on a log stored in the currently connected FR.

Parameters

index

[in] index of the desired log, starting with 0.

value

[out] pointer to the buffer which will received the returned string.

size

[in] size of the buffer pointed to by *value*.

Return Values

IdentifyLogFR returns number of bytes in the returned string, if actual length of string exceeds *size*, the string will be truncated to *size* -1 bytes. If the value specified for *index* exceeds the number of logs present in the FR (minus 1, as indexing starts with 0), *IdentifyLogFR* will return 0.

Remarks

The returned string consists of seven fields, separated by the "pipe" character ("|", 0x7C), the default log file name (including extension), log start UTC date (formatted YYYY-MM-DD, example "2000-05-12", zero padding required), log start UTC time (formatted HH:MM:SS, example "17:09:22", zero padding required), log end UTC time (formatted HH:MM:SS), pilot name, competition id, and competition class. Maximum permitted length of the returned string (excluding the terminating NUL character) is 127 characters. Example:

```
0B8X01F1.XL1|2000-11-08|20:05:21|01:21:09|J. Doe|XYZ|15M
```

Logs are indexed in descending start date/time order, the log at index 0 is the most recent log. When retrieving information on all of the logs stored within the FR, a control program should start by calling *IdentifyLogFR* with *index* 0, incrementing *index* by 1 until *IdentifyLogFR* returns 0.

DownloadLogFR

```
BOOL DownloadLogFR(DWORD index, LPCTSTR fileName, BOOL progress)
```

The *DownloadLogFR* function is used to download a log file from the currently connected FR.

Parameters

index

[in] the index of the desired log, starting with 0.

fileName

[in] a null terminated string containing the name of the file (which may include a path) to which the log will be downloaded. If NULL, the default file name will be used in the current working directory.

progress

[in] if TRUE, while the download is in progress, *DownloadLogFR* will display a modal dialog box with a progress indicator and a cancel download button. If FALSE, the download will occur silently, without a progress dialog box being displayed.

Return Values

DownloadLogFR returns TRUE if successful, FALSE if there was an error.

Remarks

If a file with the specified name and path already existed, it will be overwritten. If there is an error, *DownloadLogFR* should display a modal dialog box giving the details.

DisconnectFR

VOID DisconnectFR()

DisconnectFR is called after the control program has completed interaction with the FR, to close the communication device.

ConvertLog

BOOL ConvertLog(LPCTSTR fileName, LPCTSTR igcFileName)

[OPTIONAL] *ConvertLog* converts the log file specified by *fileName* to an IGC format file specified by *igcFileName*.

Parameters

fileName

[in] a null terminated string containing the name of an existing log file (which may include a path) in the manufacturer proprietary format.

igcFileName

[in] a null terminated string containing the name of the IGC file (which may include a path) to be created.

Return Values

Returns TRUE if successful, FALSE if there is an error.

Remarks

This is an optional function, and should only be exported by the DLL if log files are downloaded in a non-IGC proprietary format. If a file with the specified *igcFileName* already exists, it will be overwritten. If there is an error, the function should display a modal dialog with the details.

ValidateLog

BOOL ValidateLog(LPCTSTR fileName)

ValidateLog is called to authenticate the digital signature on a specified log file.

Parameters

fileName

[in] a null terminated string containing the name of an existing log file (which may include a path) to be validated.

Return Values

Returns TRUE if file can be validated, FALSE otherwise.

Remarks

If the log was not produced by a supported flight recorder, is in an unsupported format, or the digital signature is invalid, the function should display a modal dialog detailing the problem, then return FALSE.

5 PROGRAMMING FRAMEWORK FOR CONTROL PROGRAMS.

To be added later after appropriate consultation. Suggestions welcome.

----- draft ends -----

