INTERNATIONAL STANDARD

ISO 11783-13

Second edition
2011-04-01

Tractors and machinery for agriculture and forestry — Serial control and communications data network —

Part 13: File server

Tracteurs et matériels agricoles et forestiers — Réseaux de commande et de communication de données en série —

Partie 13: Serveur de fichiers

Reference number
ISO 11783-13:2011(E)

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Contents

Page

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of technical committees is to prepare International Standards. Draft International Standards adopted by the technical committees are circulated to the member bodies for voting. Publication as an International Standard requires approval by at least 75 % of the member bodies casting a vote.

ISO 11783-13 was prepared by Technical Committee ISO/TC 23, *Tractors and machinery for agriculture and forestry*, Subcommittee SC 19, *Agricultural electronics*.

This second edition cancels and replaces the first edition (ISO 11783-13:2007), of which it constitutes a minor revision.

ISO 11783 consists of the following parts, under the general title *Tractors and machinery for agriculture and forestry — Serial control and communications data network*:

— *Part 1: General standard for mobile data communication*

— *Part 2: Physical layer*

— *Part 3: Data link layer*

— *Part 4: Network layer*

— *Part 5: Network management*

— *Part 6: Virtual terminal*

— *Part 7: Implement messages application layer*

— *Part 8: Power train messages*

— *Part 9: Tractor ECU*

— *Part 10: Task controller and management information system data interchange*

— *Part 11: Mobile data element dictionary*

— *Part 12: Diagnostics services*

— *Part 13: File server*

— *Part 14: Sequence control*

Introduction

Parts 1 to 14 of ISO 11783 specify a communications system for agricultural equipment based on ISO 11898-1[1] and ISO 11898-2[2]. SAE J1939[3] documents, on which parts of ISO 11783 are based, were developed jointly for use in truck and bus applications and for construction and agriculture applications. Joint documents were completed to allow electronic units that meet the truck and bus SAE J1939 specifications to be used by agricultural and forestry equipment with minimal changes. General information on ISO 11783 is to be found in ISO 11783-1.

The purpose of ISO 11783 is to provide an open, interconnected system for on-board electronic systems. It is intended to enable electronic control units (ECUs) to communicate with each other, providing a standardized system.

The International Organization for Standardization (ISO) draws attention to the fact that it is claimed that compliance with this part of ISO 11783 may involve the use of a patent concerning the controller area network (CAN) protocol referred to throughout the document.

ISO takes no position concerning the evidence, validity and scope of this patent.

The holder of this patent has assured ISO that he is willing to negotiate licences under reasonable and non-discriminatory terms and conditions with applicants throughout the world. In this respect, the statement of the holder of this patent right is registered with ISO. Information may be obtained from:

Robert Bosch GmbH
Wernerstrasse 51
Postfach 30 02 20
D-70442 Stuttgart-Feuerbach
Germany

Attention is drawn to the possibility that some of the elements of this part of ISO 11783 may be the subject of patent rights other than those identified above. ISO shall not be held responsible for identifying any or all such patent rights.

Tractors and machinery for agriculture and forestry — Serial control and communications data network —

Part 13:
File server

1 Scope

ISO 11783 as a whole specifies a serial data network for control and communications on forestry or agricultural tractors and mounted, semi-mounted, towed or self-propelled implements. Its purpose is to standardize the method and format of transfer of data between sensors, actuators, control elements and information storage and display units, whether mounted on, or part of the tractor or implement. This part of ISO 11783 specifies the file server (FS) for use by a tractor or self-propelled implement.

2 Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO 11783-1, *Tractors and machinery for agriculture and forestry — Serial control and communications data network — Part 1: General standard for mobile data communication*

ISO 11783-3, *Tractors and machinery for agriculture and forestry — Serial control and communications data network — Part 3: Data link layer*

ISO 11783-5, *Tractors and machinery for agriculture and forestry — Serial control and communications data network — Part 5: Network management*

ISO 11783-6, *Tractors and machinery for agriculture and forestry — Serial control and communications data network — Part 6: Virtual terminal*

ISO 11783-7, *Tractors and machinery for agriculture and forestry — Serial control and communications data network — Part 7: Implement messages application layer*

3 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO 11783-1 and the following apply.

3.1
client
electronic control unit (ECU) on the mobile implement bus that uses the services of the file server

3.2
directory
file which stores administrative information about other files

3.3
file
data object that stores data on a storage device

3.4
file attribute
bit-coded information that defines the type and features of a file

3.5
file server
FS
electronic control unit (ECU) on the mobile implement bus that provides storage for files and uses a set of commands for the handling of, and access to, these files

3.6
filename
name conforming to requirements of a character set, which identifies a file or directory

NOTE See Annex A for the character set.

3.7
Handle
data object used for accessing files and directories

3.8
hidden attribute
file attribute indicating that the file should not appear in a directory listing

NOTE A client sets this attribute by using the file server (FS) commands.

3.9
path
specification of a filename that may also include the directory name

3.10
read-only attribute
file attribute used to prevent writing to, or deletion of, a file

NOTE A client sets this attribute by using the file server (FS) commands.

3.11
volume
directory that refers to a specific logical or physical storage unit or space

NOTE The primary volume is the volume used as current volume when the file server (FS) is started.

4 General

The message set specified in this part of ISO 11783 is designed to support the needs of tractors and implements in using the services of a file server (FS) — a distinct electronic control unit (ECU) on the mobile implement control system that enables all controllers to store or retrieve data from a file-based storage device.

5 Requirements

5.1 General message format

The general message format uses the parameter group number as the label for a group of parameters (see Annex B). Each of the parameters within the group can be expressed as characters, as scaled data defined by the ranges given in 5.2, or as one or more bits. Characters shall be transmitted with the left-most character first. Numerical parameters consisting of two or more data bytes shall be transmitted least significant byte (LSB) first. When variable-length messages have eight or less data bytes, these messages shall be transmitted in a single controller area network (CAN) frame. When variable length messages have nine or more data bytes, the transport protocol (TP), in accordance with ISO 11783-3, or the extended transport protocol (ETP), in accordance with ISO 11783-6, is required. When a message has less than eight data bytes, the unused bytes shall be filled with FF_{16} values.

5.2 File data format

5.2.1 Data

Data consists of a block of bytes (unsigned eight-bit values). All values in the range of 0_{10} … 255_{10}, 00_{16} … FF_{16} are allowed. There is no special handling of individual characters (control characters, end-of-line markers, end-of-file markers or similar characters).

5.2.2 Bit groups

Groups of one to eight bits are packed into one byte as bit 7 … bit 0. Groups of nine to 16 bits are packed into two bytes in the order of LSB as bit 7 … bit 0, followed by most significant byte (MSB) as bit 15 … bit 8. Unused bits in a bit group default to a value of 0 (zero).

5.2.3 Integer

Unsigned 8 bit	1 byte	$0 \ldots 2^8-1$	$0_{10} \ldots 255_{10}$
Unsigned 16 bit	2 bytes, LSB first	$0 \ldots 2^{16}-1$	$0_{10} \ldots 65535_{10}$
Unsigned 32 bit	4 bytes, LSB first	$0 \ldots 2^{32}-1$	$0_{10} \ldots 4294967295_{10}$
Signed 32 bit	4 bytes, LSB first, two's compliment	$-2^{31} \ldots 2^{31}-1$	$-2147483648_{10} \ldots +2147483647_{10}$

5.2.4 Character string

A string contains characters represented by bytes (unsigned eight-bit values). The length of a string is specified by a string length data item. Annex A specifies the characters allowed in a string used as a filename or a path name.

5.3 Data transmission control

5.3.1 General

Each communication transaction between a client and the FS is initiated by a request from the client and terminated by a response from the FS. In order to provide fail-safe communications, it is important that the client *assign* the received response to a corresponding request and repeat an erroneous request without triggering the complete execution again.

5.3.2 Strategy

The client can issue a request and receive no response because of transient communication problems. The failure can happen during the request message, i.e. the FS does not receive the request, or during the response message, i.e. the client does not receive the response. The client cannot distinguish between these two cases and shall repeat the request to obtain the requested data.

If there is no transaction strategy, the problem of the FS not receiving the request is resolved by the client sending a second request and the FS responding with the requested data. However, if the client does not receive the correct response data message and sends a second request, the FS then sends the next data from the file; this is because a data request automatically advances to the next data in the file.

A transaction strategy is therefore required to prevent such errors. Each client on the network maintains its own transaction number (TAN) counter, which should start at 0 after a power cycle.

Each client generates a TAN for each request that it sends to the FS. This is done by incrementing the last TAN used for the next request. The client is responsible for checking that a received response contains the same TAN that was used in the request during the communication session, thus ensuring that there are no lost commands. The FS shall remember the last command processed and response message sent for each client. This is done by incrementing the last TAN for the next request. The FS compares each new request with the previous request from the same client. If the TAN is not the same, the request is implemented and the response is sent. If the TAN is the same as the previously received request, the request is not implemented and the previous response is sent. Thus if the client sends a second request, in the case where the FS never received the first request, the FS *receives* the TAN for the first time, *implements* the request and sends the correct data response. If the FS receives a request with the same TAN that it has already received, it does not implement the request, but the previous response is retransmitted.

5.3.3 Timeout

The execution time of all FS commands (the time between request and response) is maintained within reasonable limits. The client shall monitor the time while waiting for a response.

The timeouts specified in ISO 11783-3 for the transport protocol and in ISO 11783-6 for the extended transport protocol shall be met for the execution of commands.

If a timeout expires, the request is assumed to have failed and the client shall repeat the request using the same TAN.

If a request response takes longer than 200 ms after the completion of the request, the FS shall send the status message to indicate busy state to the client. This provides a request timeout of 600 ms if the FS status message does not show a busy status.

5.4 Date and time support

Several FS commands require a file date and time. UTC[1)] is used for this time. The file server's implementation of real time support can be either by maintaining its own real time information or by requesting the time and date information using the Time/Date parameter group specified in ISO 11783-7. The date and time of a file is the latest date and time when a file was actually modified. A file which is opened for read/write access but is not modified by a write action shall not get a later date and time.

5.5 Multi-client support

The file server shall support one or more clients. If more than one client has a connection simultaneously, the FS shall function with each client as if it is the only one on the network. There shall be no interference between the commands processed for different clients.

---

1) Coordinated universal time, or universal time, formerly known as Greenwich mean time (GMT).

Upon connection of a client, the file server initiates the current directory for that client as the root directory of the primary volume of the FS file system. If there are no volumes, then the current directory is set to the list of volumes "\\". The client is required to use the appropriate Change Current Directory or Open File commands to access files that need to be unique for that client. In the case where multiple clients require access to common files, these clients are responsible for synchronizing their directory and file naming conventions to enable access to these common files. To prevent unintentional access to manufacturer proprietary files, a reserved directory name containing the manufacturer code according to ISO 11783-5 is specified. The naming convention of the manufacturer-specific directory is:

MCMC0000

where 0000 contains the four-digit manufacturer code in decimal representation, formatted with leading zeroes. A client shall not use this manufacturer-coded directory name using a manufacturer code other than the manufacturer code in its NAME field. When the client attempts to open a file in a manufacturer-specific directory where the manufacturer code in the NAME of that client is not that of the manufacturer-specific directory name, the FS shall prevent access and return an "access denied" error code.

When a file server supports multiple volumes, manufacturer-specific directories can be created on each volume. Creation of a manufacturer-specific directory is the responsibility of the client. The manufacturer-specific directory shall only be placed at the root of each volume.

5.6 File Handles

An FS may support multiple file Handles. Many of the commands available for the FS create and/or use file Handles. However, there are some commands that only use folder or filenames. Internally, if the FS creates a file Handle to process these commands, the number of open files shall be incremented to reflect the internal status of the FS.

5.7 Volumes

Different types of media (FLASH, removable media, ruggedized disk drives) can be assigned to different volumes.

An FS can support multiple volumes. It is also possible for an FS to list no volumes — for example, with uninitialized media or no device found.

The list of volumes, specified by "\\", is the highest layer (or base) of a directory structure.

A special service tool for the FS can be used to create volumes as specified in Annex C, Initialize Volume Request (C.5.2.2) message. The names of the volumes are determined by the FS; however, the FS can allow a service tool to name them as specified for this message.

NOTE This part of ISO 11783 does not specify *how* the service tool selects the media or volumes to initialize if they are not named and listed in the list of volumes "\\".

Annex A
(normative)

Character set

A.1 Valid characters

The file server uses filenames and path names. Every character used for one of the 8.3 names (see below) is validated by the FS using the appropriate subset of Table A.1 (based on ISO/IEC 8859-1) and the filename and path definitions given in A.2. Only printable characters are visible when presenting the filename or path name to a user. For case-insensitive file systems, the FS shall convert the lower-case characters (61_{16} to $7A_{16}$) to upper-case (41_{16} to $5A_{16}$). File servers that do not support long filenames shall use an 8.3 name and extension notation where the name is a maximum of eight characters long, optionally followed by an extension that starts with a single dot ($2E_{16}$), and finishes with a maximum of three characters. File servers that support long filenames shall use the filename and path defined in A.2.

Table A.1 — ISO Latin 1 character set

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	✂	✂	✂	✂	✂	✂	✂	✂	✂	✂	✂	✂	✂	✂	✂	✂
1	✂	✂	✂	✂	✂	✂	✂	✂	✂	✂	✂	✂	✂	✂	✂	✂
2	space	!	“	#	$	%	&	'	(	)	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	✂
8	✂	✂	✂	✂	✂	✂	✂	✂	✂	✂	✂	✂	✂	✂	✂	✂
9	✂	✂	✂	✂	✂	✂	✂	✂	✂	✂	✂	✂	✂	✂	✂	✂
A	✂	¡	¢	£	¤	¥	¦	§	¨	©	«	«	¬	-	®	¯
B	°	±	²	³	³	µ	¶	·	¸	¹	º	»	¼	½	¾	¿
C	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
D	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
E	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
F	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ

✂ non-printable character

NOTE This information is given for reference; the following does not use all the available characters given here.

A.2 Filename and path definitions

A.2.1 General

Definitions:

[ ] any of the characters in Table A.1, including none from the set (optional);

[A-B] defines an inclusive range from the first through the last;

( ) group;

< > character class;

\ escapes the following character, as in "\[", which indicates a single left bracket, not the containment of a set;

A | B sequence "A" or "B";

A + B sequence of A followed by B;

{m} exactly m of the preceding set;

{m,n} from m up to and including n of the preceding set;

\xXX character code in hexadecimal notation where XX are two hexadecimal digits (\x20, for example, indicates character code 32, which is the space character).

A.2.2 Name definitions

A.2.2.1 Names

Names are from one to 254 characters in length, using the character set given below. The commonly used file systems given in Annex D were used to determine name restrictions that would allow these names to be used with minimal feature loss.

ShortNameChar ::= [0-9A-Z\-!#$%&'\(\)@^_`\{\}~\xA1-\xFF] {1}

LongNameChar ::= any single character defined by Unicode/ISO/IEC 10646 except NUL, \, *, or ?

WildCardChar ::= [*?] {1}

ManufacturerSpecificDirectoryChar ::= [~] {1}

PathSeparatorChar ::= [\\ ] {1}

VolumeListIndicator ::= [\\ ] {2}

ParentFolderIndicator ::= [.] {2}

CurrentFolderIndicator ::= [.] {1}

MfgSpecificFolderIndicator ::= [<ManufacturerSpecificDirectoryChar>] {1}

ShortWildCardNameChar::= [<ShortNameChar> | <WildCardChar>] {1}

LongWildCardNameChar::= [<LongNameChar> | <WildCardChar>] {1}

ShortName ::= [<ShortNameChar>] {1,8} [.+ [<ShortNameChar>] {0,3}]

LongName ::= [<LongNameChar>] {1,254}

ShortWildCardName ::= [<ShortWildCardNameChar>] {1,8} [.+ [<ShortWildCardNameChar>] {0,3}]

LongWildCardName ::= [<LongWildCardNameChar>] {1,254}

A.2.2.2 Filenames

Filenames use the names defined in A.2.2.1.

ShortFileName ::= <ShortName>

LongFileName ::= <LongName>

EXAMPLE Test, Test.txt, Test Filename.long.name (specifically a LongName).

A.2.2.3 Volumes

Volumes use the names defined in A.2.2.1.

VolumeName ::= <LongName>

EXAMPLE VOL_B, Flash Volume (specifically a LongName).

A.2.3 Path definitions

A.2.3.1 General

A path definition is similar to a filename definition but has additional prefix definitions and delimiters between path segments.

When a directory listing from path "\\" (two backslashes) is requested, the FS shall return a list of volumes. All file servers shall support "\\" so that the clients can query the volumes (including removable media), even if the FS has only one volume.

The two predefined special directory names, "." and "..", refer to the current (".") and parent ("..") directories. These predefined directory names shall not be reported in a directory listing but may be used in a path name to specify reference to a current or parent directory.

The "~" character (*tilde*) may be used as a placeholder for the manufacturer-specific directory of a client. This character can only be specified at the beginning of a path or after a volume name and shall be replaced by the FS with the manufacturer-specific directory name on the current volume. If there is no current volume, then the server primary volume shall be used. The "~" can be used in the name, but cannot be the only character, since this would be interpreted as the manufacturer-specific directory. For example, "~\file1.txt" to "MCMC0000\file1.txt".

ShortFolderName ::= [<ShortName> | <ParentFolderIndicator> | <CurrentFolderIndicator>] {1}

LongFolderName ::= [<LongName> | <ParentFolderIndicator> | <CurrentFolderIndicator>] {1}

ShortPathName ::= [

[<VolumeListIndictor>] |

[[<VolumeListIndictor>] + <VolumeName> + <PathSeparatorChar> + [<MfgSpecificFolderIndicator> + <PathSeparatorChar>] {0,1} + [<ShortFolderName> + <PathSeparatorChar>] {0,n}] |

[[<PathSeparatorChar>] {0,1} + [<ShortFolderName> + <PathSeparatorChar>] {0,n}] |

[[<MfgSpecificFolderIndicator> + <PathSeparatorChar>] {0,1} + [<ShortFolderName> + <PathSeparatorChar>] {0,n}]

] {1}

LongPathName ::= [

[<VolumeListIndictor>] |

[[<VolumeListIndictor>] + <VolumeName> + <PathSeparatorChar> + [<MfgSpecificFolderIndicator> + <PathSeparatorChar>] {0,1} + [<LongFolderName> + <PathSeparatorChar>] {0,n}] |

[[<PathSeparatorChar>] {0,1} + [<LongFolderName> + <PathSeparatorChar>] {0,n}] |

[[<MfgSpecificFolderIndicator> + <PathSeparatorChar>] {0,1} + [<LongFolderName> + <PathSeparatorChar>] {0,n}]

] {1}

EXAMPLE 1 Path relative to current directory:

.\

..\path\

..\Long path name\ (specifically a LongName)

Path\

Level1\Level2\

Path.dir\

Long path name\ (specifically a LongName)

EXAMPLE 2 Path relative to root of current volume:

\Path\

\Level1\Level2\

\Path.dir\

\Long path name\ (specifically a LongName)

Path including volume:

\\VOL_B\path\

\\VOL_B\Level1\Level2\

\\Flash Volume\Long path name\ (specifically a LongName)

EXAMPLE 3 Path using manufacturer-specific folder indicator:

~\

~\Path\

~\Level1\Level2\

\\VOL_B\~\path\

\\VOL_B\~\Level1\Level2\

A.2.3.2 Path and filename

This path name includes as much path information as needed to produce an unambiguous specification of the path to the file:

ShortPathAndFileName ::= [

[[<VolumeListIndictor>] + <VolumeName> + <PathSeparatorChar> + [<MfgSpecificFolderIndicator> + <PathSeparatorChar>] {0,1} + [<ShortFolderName> + <PathSeparatorChar>] {0,n} + [<ShortFileName>]] |

[[<PathSeparatorChar>] {0,1} + [<ShortFolderName> + <PathSeparatorChar>] {0,n} + [<ShortFileName>]] |

[[<MfgSpecificFolderIndicator> + <PathSeparatorChar>] {0,1} + [<ShortFolderName> + <PathSeparatorChar>] {0,n} + [<ShortFileName>]]

] {1}

LongPathAndFileName ::= [

[[<VolumeListIndictor>] + <VolumeName> + <PathSeparatorChar> + [<MfgSpecificFolderIndicator> + <PathSeparatorChar>] {0,1} + [<LongFolderName> + <PathSeparatorChar>] {0,n} + [<LongFileName>]] |

[[<PathSeparatorChar>] {0,1} + [<LongFolderName> + <PathSeparatorChar>] {0,n} + [<LongFileName>]] |

[[<MfgSpecificFolderIndicator> + <PathSeparatorChar>] {0,1} + [<LongFolderName> + <PathSeparatorChar>] {0,n} + [<LongFileName>]]

] {1}

EXAMPLE 1 Path relative to current directory:

Test.txt

path\Test.txt

Long path name\Test Filename.long.name (specifically a LongName)

EXAMPLE 2 Path relative to root of current volume:

\path\Test

~\path\Test

~\Level1\Level2\Test

EXAMPLE 3 Path including volume:

\\VOL_B\path\Test

\\VOL_B\~\path\Test

\\VOL_B\path\Test.txt

\\VOL_B\Level1\Level2\Test

\\Flash Volume\Long path name\Test Filename.long.name (specifically a LongName)

Figure A.1 shows an example of an FS architecture with volumes, directories and files, and an example path name. The example path is the manufacturer-specific directory that shall only be used by clients with a matching manufacturer code in their NAME field.

A.2.3.3 Search definitions

The wildcards "*" and "?" may be used:

— "*" is a wildcard for 0 or more characters of a filename or folder name;

— "?" is a wildcard for a single character in a filename or folder name. Wildcards shall only be used for directory listings.

ShortPathAndWildCardName ::= [

[[<VolumeListIndictor>] + <VolumeName> + <PathSeparatorChar> + [<MfgSpecificFolderIndicator> + <PathSeparatorChar>] {0,1} + [<ShortFolderName> + <PathSeparatorChar>] {0,n} + <ShortWildCardName>] |

[[<PathSeparatorChar>] {0,1} + [<ShortFolderName> + <PathSeparatorChar>] {0,n} + <ShortWildCardName>] |

[[<MfgSpecificFolderIndicator> + <PathSeparatorChar>] {0,1} + [<ShortFolderName> + <PathSeparatorChar>] {0,n} + <ShortWildCardName>]

] {1}

LongPathAndWildCardName ::= [

[[<VolumeListIndictor>] + <VolumeName> + <PathSeparatorChar> + [<MfgSpecificFolderIndicator> + <PathSeparatorChar>] {0,1} + [<LongFolderName> + <PathSeparatorChar>] {0,n} + <LongWildCardName>] |

[[<PathSeparatorChar>] {0,1} + [<LongFolderName> + <PathSeparatorChar>] {0,n} + <LongWildCardName>] |

[[<MfgSpecificFolderIndicator> + <PathSeparatorChar>] {0,1} + [<LongFolderName> + <PathSeparatorChar>] {0,n} + <LongWildCardName>]

] {1}

EXAMPLE 1 Path relative to current directory:

*

?ath

?a*

~*

~\?ath

EXAMPLE 2 Path relative to root of current volume:

*

\?ath

~\path*

EXAMPLE 3 Path including volume:

\\VOL_B*

\\VOL_B\?ath

\\VOL_B\path\Test*

\\VOL_B\path\Test*.txt

\\VOL_B\Level1\Level2\Test.*

\\VOL_B\Level1\Level2\T?st.txt

\\Flash Volume\Long path name\Test ???? Name.long.name (specifically a LongName)

\\Flash Volume\Long path name\Test * Name.*.name (specifically a LongName)

\\Flash Volume\Long path name\T?st Filename.long.name (specifically a LongName)"

Figure A.1 shows an example of FS architecture with volumes, directories and files, and an example path name.

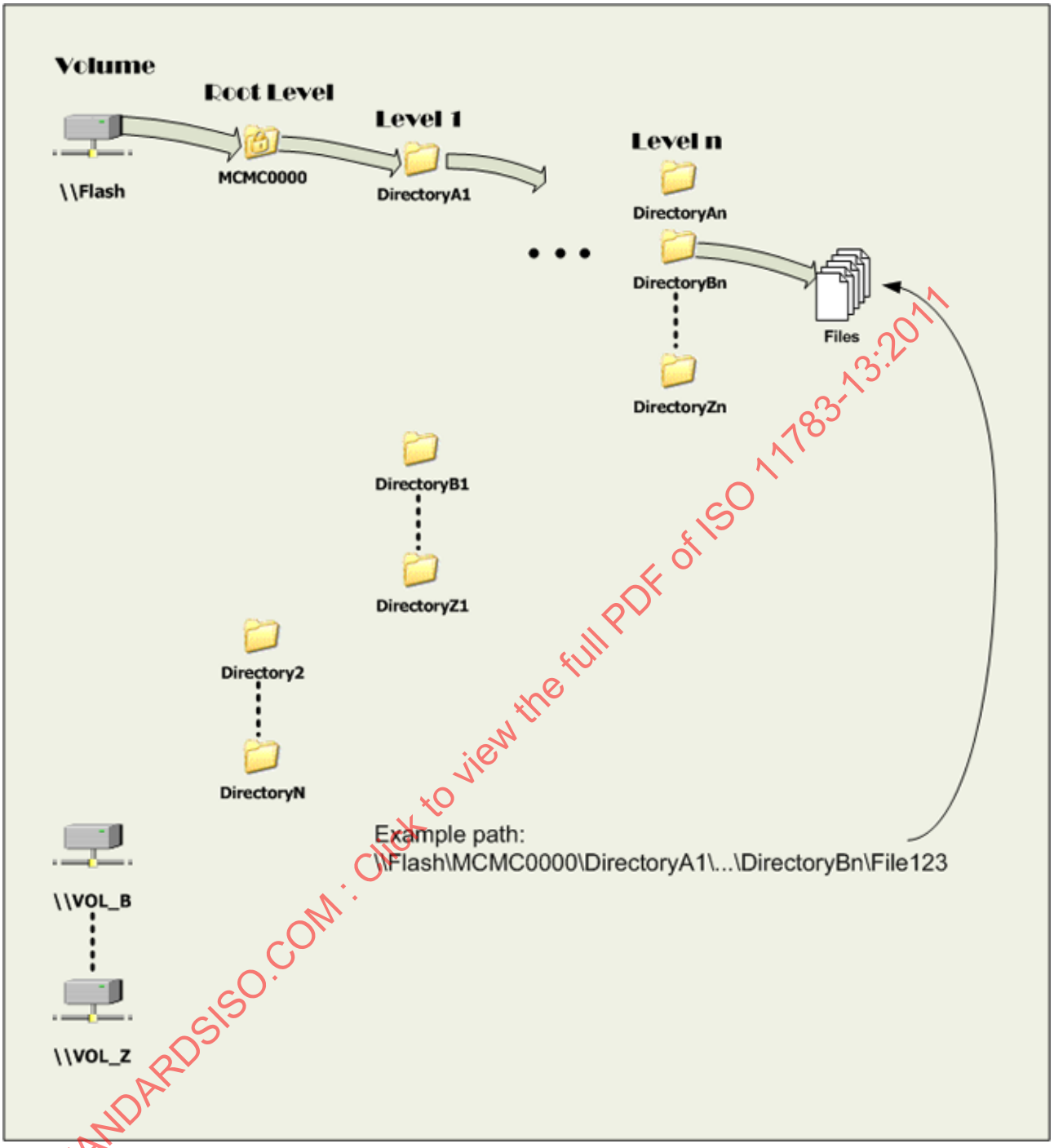

Figure A.1 — Example file server architecture

Annex B
(normative)

Parameter definitions

B.1 Command groups

File server commands are divided into groups; four bits specify the group of the commands.

Data length: 4 bits

Value	Meaning
0000	Connection Management
0001	Directory Handling
0010	File Access
0011	File Handling
0100	Volume Handling

B.2 Command functions

Each FS command group has a number of functions. The lower four bits of a command byte specify the function of the command.

Data length: 4 bits

Value	Meaning
0_{16} … F_{16}	Defined in each command message

B.3 File Server Status

The current status of the FS.

Data length: 1 byte

Bit	Value	Meaning
7 … 2	000000	Reserved, send as 000000
1	1	Busy writing
0	1	Busy reading

B.4 Number of Open Files

The number of files that are currently open at the FS.

Data length: 1 byte

Resolution: 1 bit

Data Range: 0_{10} … 255_{10} (unsigned 8 bits)

B.5 Version Number

The number of the edition or version of ISO 11783-13 with which the FS or client is compliant.

The Version Number parameter reported by the client shall reflect the edition (version) of the International Standard (i.e. this part of ISO 11783) to which the client is designed. It shall not change at runtime due to adaptations to different file servers. For example, a Version 3 client would still report Version 3 behaviour in this parameter, even if the client falls back to Version 2 behaviour to communicate to a Version 2 file server. The FS may choose to report this or provide it for diagnostics, but shall not reject communication or the request based on the reported Version Number.

Data length: 1 byte

Value	Meaning
0_{10}	Draft edition of the International Standard
1_{10}	Final draft edition of the International Standard
2_{10}	First published edition of the International Standard
3_{10}	Second published edition of the International Standard
4_{10} ... 254_{10}	Reserved
255_{10}	Compliant with Version 2 and prior (client only)

B.6 Maximum Number of Simultaneously Open Files

The maximum number of files that can be opened simultaneously at the FS.

Data length: 1 byte

Resolution: 1 bit

Data Range: 0_{10} … 255_{10} (unsigned 8 bits)

B.7 File Server Capabilities

Data length: 1 byte

Bit	Value	Meaning
7 .. 1	0000000	Reserved, send as 0000000
0	1	File server supports multiple volumes

B.8 Transaction Number

A number (TAN) assigned to a request so that the corresponding response can be identified.

Data length: 1 byte

Resolution: 1 bit

Data Range: 0 to 255

B.9 Error Code

The error code returned in an FS to a client response message. When the error code in the response message is unequal to 0_{10} "Success", the other parameters after the error code in the response message can contain incorrect data and shall be ignored by the client. (Version 3 and later FS support this error code.)

Data length: 1 byte

Value	Meaning
0_{10}	Success
1_{10}	Access Denied
2_{10}	Invalid Access
3_{10}	Too many files open
4_{10}	File, path or volume not found
5_{10}	Invalid Handle
6_{10}	Invalid given source name
7_{10}	Invalid given destination name
8_{10}	Volume out of free space
9_{10}	Failure during a write operation
10_{10}	Media is not present [formerly the code for error 13, below, in Version 2 FSs (see B.5)]
11_{10}	Failure during a read operation
12_{10}	Function not supported
13_{10}	Volume is possibly not initialized
14_{10} … 41_{10}	Reserved
42_{10}	Invalid request length (used when the file pointer hits the start/top of the file or on invalid space request of the volume)
43_{10}	Out of memory (used by FS to indicate out of resources at this time and cannot complete request)
44_{10}	Any other error
45_{10}	File pointer at end of file
46_{10} … 255_{10}	Reserved

B.10 Handle

The data object assigned by the FS and used by the FS and client to reference a file or directory for requested actions.

Data length: 1 byte

Value	Meaning
0_{10} … 254_{10}	Value of the Handle assigned by the FS for further access to a file
255_{10}	Error when assigning a Handle for a file

B.11 Space

The space in units of 512.

Data length: 4 bytes

Resolution: 1 bit

Data Range: 0_{10} … 4294967295_{10} (unsigned 32 bits)

B.12 Path Name Length

Number of characters in a path name. This can be a volume, path, filename, wildcard, or combination thereof.

Data length: 2 bytes

Resolution: 1 bit

Data Range: 0_{10} … 65535_{10} (unsigned 16 bits)

B.13 Path Name

The allowed characters in a path name are specified in A.2.3.1.

B.14 Flags

The data object used by a client to specify the mode of file or directory access requested.

Data length: 1 byte

Bit	Value	Meaning
7 … 5	000	Reserved, send as 000
4	0	Open file for shared read access
	1	Open file with exclusive access (fails if already open)
3	0	Open file for random access (file pointer set to the start of the file)
	1	Open file for appending data to the end of the file (file pointer set to the end of the file)
2	0	Open an existing file (fails if non-existent file)
	1	Create a new file and/or directories if not yet existing
1,0	00	Open file for reading only
	01	Open file for writing only
	10	Open file for reading and writing
	11	Open directory

When bits 1 and 0 are set for “Open directory”, bit 3 is ignored and treated as read-only, because used only to query what is in the directory. A directory path is created using “Open directory” and “Create a new file if not yet existing”.

B.15 Attributes

The data object used by the FS to describe the file to the client.

Data length: 1 byte

Bit	Value	Meaning
7	0	Volume is case-insensitive
	1	Volume is case-sensitive (Version 3 and later FS support this attribute)
6	0	Volume is removable
	1	Volume is not removable
5	0	Volume does not support long filenames
	1	Volume supports long filenames
4	0	Handle does not specify a directory
	1	Handle specifies a directory
3	0	Handle does not specify a volume
	1	Handle specifies a volume
2	0	Volume does not support hidden attribute
	1	Volume supports hidden attribute (Version 3 and later FS support this attribute)
1	0	“Hidden” attribute is not set
	1	“Hidden” attribute is set (not applicable unless volume supports hidden attribute)
0	0	“Read-only” attribute is not set
	1	“Read-only” attribute is set

B.16 Set Attributes Command

The command from the client to the FS for setting or clearing file attributes.

Data length: 1 byte

Bit	Value	Meaning
7,6	11	Reserved, set to 11
5,4	11	Reserved, set to 11
3,2	00	Clear “hidden” attribute
	01	Set “hidden” attribute (not applicable unless volume supports hidden attribute)
	11	Don't care, leave “hidden” attribute in current state
1,0	00	Clear “read-only” attribute
	01	Set “read-only” attribute
	11	Don't care, leave “read-only” attribute in current state

B.17 Position Mode

The position mode specifies the location from which the offset value is used to determine the file pointer position.

Data length: 1 byte

Value	Meaning
0_{10}	From the beginning of the file
1_{10}	From the current pointer position
2_{10}	From the end of the file
3_{10} … 255_{10}	Reserved

B.18 Offset

The offset is used with the position mode to determine the file pointer position.

Data length: 4 bytes

Resolution: 1 bit

Data Range: -2147483648_{10} … $+2147483647_{10}$ (signed 32 bits)

B.19 Position

The value of the file pointer position.

Data length: 4 bytes

Resolution: 1 bit

Data Range: 0_{10} … 4294967295_{10} (unsigned 32 bits)

B.20 Count

The number of bytes of data requested to be read from, or written to, a file, or the number of directory entries read from a directory.

Data length: 2 bytes

Resolution: 1 bit

Data Range: 0_{10} … 65535_{10} (unsigned 16 bits)

B.21 Directory Entry

Data length: Variable

Byte 1	Filename Length	see B.22
Byte 2–n	Filename	see B.23
Byte n + 1	Attributes	see B.15
Bytes n + 2, n + 3	File Date	see B.24
Bytes n + 4, n + 5	File Time	see B.25
Bytes n + 6 … n + 9	Size	see B.26

B.22 Filename Length

Number of characters in a filename.

For an FS without support for long filenames, the filename length is a maximum of 1 ... 12 characters.

For an FS with support for long filenames, the filename is a maximum of 1 ... 254 characters. (Version 2 and prior specified a maximum of 31 characters.)

Data length: 1 byte

Resolution: 1 bit

Data Range: 1_{10} … 254_{10} (unsigned 8 bits)

B.23 Filename

The allowed characters in a filename are specified in A.2.2.2.

B.24 File Date

The file date is represented by a bit group with the following 16 bits encoding.

Data length: 2 bytes

Bits 15 … 9	0 … 127	Year – 1980 (difference between the year and 1980)
Bits 8 … 5	1 … 12	Month (1 = January, …, 12 = December)
Bits 4 … 0	1 … 31	Day

If a file date is not available in an implementation, all bits are set to zero ($0_{10} = 0000_{16}$) resulting in "1980-00-00".

B.25 File Time

The file time is represented by a bit group with the following 16 bits encoding.

Data length: 2 bytes

Bits 15 … 11	0 … 23	Hours
Bits 10 … 5	0 … 59	Minutes
Bits 4 … 0	1 … 29	Seconds (in steps of 2 s)

If a file time is not available in an implementation, all bits are set to zero ($0_{10} = 0000_{16}$) resulting in "00-00-00".

B.26 Size

The number of bytes in a file.

Data length: 4 bytes

Resolution: 1 bit

Data range: 0_{10} … 4294967295_{10} (unsigned 32 bits)

B.27 File Handling Mode

Data length: 1 byte

Bit	Value	Meaning
7–3	00000	Reserved, send as 00000
2	0	“Recursive” mode is not set
	1	“Recursive” mode is set
1	0	“Force” mode is not set
	1	“Force” mode is set
0	0	“Copy” mode is not set
	1	“Copy” mode is set

B.28 Report Hidden Files

Data length: 1 byte

Value	Meaning
0_{10}	Do not report hidden files in a directory listing
1_{10}	Report hidden files in a directory listing
2_{10} ... 254_{10}	Reserved
255_{10}	Parameter is not available, FS shall not report hidden files in a directory listing

B.29 Volume Flags

The data object used by the client to specify the mode of volume access requested.

Data length: 1 byte

Bit	Value	Meaning
7 … 2	000000	Reserved, send as 000000
1	0	Create volume using all available space
	1	Create volume using space specified
0	0	Create a new volume if not yet existing (fails if existing volume)
	1	Overwrite the existing volume

B.30 Volume Mode

The data object used by a client to specify the mode of the volume access requested. A value of 00000000 requests the current status. (This parameter applies for Version 3 and later FS.)

Data length: 1 byte

Bit	Value	Meaning
7 … 2	000000	Reserved, send as 000000
1	1	Request volume to prepare for removal
0	1	Report volume in use by client
	0	Report volume not in use by client

B.31 Volume Status

The current status of the volume. (This parameter applies for Version 3 and later FS.)

Data length: 1 byte

Bit	Value	Meaning
7 … 3	00000	Reserved, send as 00000
2 … 0	7 … 4	Reserved
	3	Removed
	2	Preparing for removal
	1	In use
	0	Present

B.32 Maximum Time before Volume Removal

Maximum time a volume can be held off from being removed. (This parameter applies for Version 3 and later FS.)

Data length: 1 byte
Resolution: 1 min/bit, 0 offset
Data range: 0 to 250
Type: Measured

B.33 Volume Name

The allowed characters in a volume name are specified in A.2.2.3.

B.34 Volume, Path, File and Wildcard Name

The allowed characters in a volume, path, file and wildcard name are specified in A.2.2.3, A.2.3.1, A.2.3.2 and A.2.3.3, respectively.

B.35 Volume, Path and Filename

The allowed characters in a volume, path and file name are specified in A.2.2.3, A.2.3.1 and A.2.3.2, respectively.

Annex C
(normative)

File server message definitions

C.1 File server messages and data transfer

C.1.1 Overview

Two PGN (parameter group numbers) are reserved for the FS message protocol:

a) File Server to Client

Transmission repetition rate:	As required
Data length:	Variable
Data Page field:	0
PDU Format field:	171
PDU specific field:	Destination address
Default priority:	7
Parameter group number:	43776 ($AB00_{16}$)

b) Client to File Server

Transmission repetition rate:	As required
Data length:	Variable
Data Page field:	0
PDU Format field:	170
PDU specific field:	Destination address
Default priority:	7
Parameter group number:	43520 ($AA00_{16}$)

Before a client (ECU) begins to maintain a connection with the FS, it can obtain information about the file server's capabilities using the above-listed PGN. Those PGN are also used to transfer the data to or from the FS utilizing the transport protocol specified in ISO 11783-3 or the extended transport protocol given in ISO 11783-6. Destination-specific messages shall be used and connection management implemented.

The client (ECU) shall wait for a response before sending another command. A fixed timeout for reception of the response cannot be given due to the fact that various commands can have very different response times. Instead, FS command processing can be monitored by the client through the File Server Status message.

The error code 12_{10}, “Function not supported”, enables the FS to indicate to a client that a specific function is not supported. All file servers shall implement the complete set of functions; nevertheless, an FS based on a certain revision of this part of ISO 11783 might support fewer functions in the case of the addition of new functions in a later revision. Each unsupported function in the command group's directory handling, file access, file handling and volume handling shall obtain a response from an FS with the error code set to “Function not supported” when the requested function is not available in the file server.

C.1.2 File Server Status

The File Server Status message is sent by the FS to provide file server status information. A timeout of 6 s on this message indicates a possible shutdown of the FS, thus enabling the client to free allocated FS communication resources.

Transmission repetition rate:	2 000 ms when the status is not busy, 200 ms when the status is busy reading or writing and, on change of byte 2, up to five messages per second			
Data length:	8 bytes			
Parameter group number:	FS to client, destination-specific, use global address: FF_{16}			
Byte 1	FS function = 0_{10}			
	Bits 7–4 0000	Command	Connection Management	see B.1
	Bits 3–0 0000	Function	File Server Status	see B.2
Byte 2		File Server Status		see B.3
Byte 3		Number of Open Files		see B.4
Bytes 4–8		Reserved, transmit as FF_{16}		

C.1.3 Client Connection Maintenance

The Client Connection Maintenance message is sent by a client in order to maintain a connection with the FS. The client sends this message when actively interacting with the FS. When this message is no longer received by the FS for 6 s, the open files are closed and all Handles for that client become invalid. The client's working directory is also lost and set back to the default. This mechanism is necessary to allow the FS to close resources correctly when a client is disconnected from the CAN bus without proper closing of any open Handles.

Transmission repetition rate:	2 000 ms			
Data length:	8 bytes			
Parameter group number:	Client to FS, destination-specific			
Byte 1	FS function = 0_{10}			
	Bits 7–4 0000	Command	Connection Management	see B.1
	Bits 3–0 0000	Function	Client Connection Maintenance	see B.2
Byte 2		Version Number		see B.5
Bytes 3–8		Reserved, transmit as FF_{16}		

C.1.4 Get File Server Properties

The Get File Server Properties message is sent by the client to request the FS properties.

Transmission repetition rate:	On request
Data length:	8 bytes

Parameter group number:		Client to FS, destination-specific		
Byte 1	FS function = 1_{10}			
	Bits 7–4 0000	Command	Connection Management	see B.1
	Bits 3–0 0001	Function	Get File Server Properties	see B.2
Bytes 2–8		Reserved, transmit as FF_{16}		

C.1.5 Get File Server Properties Response

The Get File Server Properties Response message is sent by the FS to a client in response to the Get File Server Properties message.

Transmission repetition rate:		In response to Get File Server Properties message		
Data length:		8 bytes		
Parameter group number:		FS to client, destination-specific		
Byte 1	FS function = 1_{10}			
	Bits 7–4 0000	Command	Connection Management	see B.1
	Bits 3–0 0001	Function	Get File Server Properties	see B.2
Byte 2		Version Number		see B.5
Byte 3		Maximum Number of Simultaneously Open Files		see B.6
Byte 4		File Server Capabilities		see B.7
Bytes 5–8		Reserved, transmit as FF_{16}		

C.1.6 Volume Status Request

The Volume Status Request message is sent by the client to command the file server volume status or request the current volume status. Path Name Length of zero shall be used to request the volume of the current directory of the client. (This message applies for Version 3 and later FS.)

Transmission repetition rate:		On request		
Data length:		Variable		
Parameter group number:		Client to FS, destination-specific		
Byte 1		FS function = 2_{10}		
	Bits 7–4	0000	Command Connection Management	see B.1
	Bits 3–0	0010	Function Removable Media Status	see B.2
Byte 2		Volume Mode		see B.30
Byte 3, 4		Path Name Length		see B.12
Bytes 5–*n*		Volume Name		see B.34

C.1.7 Volume Status Response

The Volume Status Response message is sent by the file server to a client in response to the Volume Status Request or on volume status change. On volume status changes, the FS sends this message to the global address (GA). If a client request causes the status to change, then the FS shall send the response to the GA, so that all clients are informed. Since the global response is on status change, this implies the error code will be Success; therefore, non-success shall only be sent to the requester. If the client "Request volume to prepare for removal" and the volume requested is not removable, or the Path Name Length of request is zero and the current directory is not set, then the FS shall return "Invalid Access". The filer server can use a Volume Name of "\\" to indicate a status change of all volumes on the FS.

Once the FS detects a removable volume is present, it shall report that the volume is "Present".

If a "Request volume to prepare for removal" is sent by a client, the FS shall report that the volume is "Preparing for removal" and all clients shall close their files and/or directories. In order to keep the volume at "Preparing for removal", the client needs to continually send a "Maintain" request.

The volume status transition from "Preparing for removal" to "Removed" is sent to all clients once they have all closed their files and/or directories. The FS maintains the volume for 2 s after the last client request of "Maintain" (by client reporting that the volume is in use within the Volume Status Request) for a "Maximum Time Before Volume Removal". The latter is determined by the FS and could be the time presented to the user when ejecting the media or the time allowed at power down that the FS can be held on. If all clients report the volume is no longer in use, the FS can remove the volume immediately.

If a client requests the status of the volume and there is a file or directory open on the volume or there has been a "Maintain" request on a "Preparing for removal", then the FS shall report the volume as "In use".

(This message applies for Version 3 and later FS.)

Transmission repetition rate:		On request and on change of Volume Status	
Data length:		Variable	
Parameter group number:		FS to client, destination-specific or use global address: FF_{16}	
Byte 1	FS function = 2_{10}		
Bits 7–4	0000	Command Connection Management	see B.1
Bits 3–0	0010	Function Volume Status	see B.2
Byte 2		Volume Status	see B.31
Byte 3		Maximum Time Before Volume Removal	see B.32
Byte 4		Error Code	see B.9
		0_{10} Success	
		1_{10} Access denied	
		2_{10} Invalid Access	
		4_{10} File, path or volume not found	
		6_{10} Invalid given source name	
		43_{10} Out of memory	
		44_{10} Any other error	
Byte 5, 6		Path Name Length	see B.12
Byte 7–*n*		Volume Name	see B.34

C.2 Directory Handling

C.2.1 Overview

Directory Handling consists of commands to get or set the current directory, which is used for requests where the path argument specifies no directory.

C.2.2 Get Current Directory

C.2.2.1 General

Get Current Directory returns the current directory as a pathname. After successful completion of the Get Current Directory Request, the full path in the form "\\VOL\DIR\SUBDIR" is reported.

C.2.2.2 Get Current Directory Request

Transmission repetition rate:		On request		
Data length:		8 bytes		
Parameter group number:		Client to FS, destination-specific		
Byte 1	FS function = 16_{10}			
	Bits 7–4 0001	Command	Directory Handling	see B.1
	Bits 3–0 0000	Function	Get Current Directory	see B.2
Byte 2		TAN		see B.8
Bytes 3–8		Reserved, transmit as FF_{16}		

C.2.2.3 Get Current Directory Response

Transmission repetition rate:		In response to Get Current Directory Request message		
Data length:		Variable		
Parameter group number:		FS to client, destination-specific		
Byte 1	FS function = 16_{10}			
	Bits 7–4 0001	Command	Directory Handling	see B.1
	Bits 3–0 0000	Function	Get Current Directory	see B.2
Byte 2		TAN		see B.8
Byte 3		Error Code		see B.9
		0_{10}	Success	
		10_{10}	Media is not present [formerly the code for error 13, below, in Version 2 FSs (see B.5)]	
		13_{10}	Volume is possibly not initialized	
		43_{10}	Out of memory	
		44_{10}	Any other error	
Bytes 4–7		Total Space (in units of 512 bytes)		see B.11
Bytes 8–11		Free Space (in units of 512 bytes)		see B.11
Bytes 12,13		Path Name Length		see B.12
Bytes 14–*n*		Path Name		see B.13

C.2.3 Change Current Directory

C.2.3.1 General

Change Current Directory selects the current directory.

C.2.3.2 Change Current Directory Request

Transmission repetition rate:		On request		
Data length:		Variable		
Parameter group number:		Client to FS, destination-specific		
Byte 1	FS function = 17_{10}			
	Bits 7–4 0001	Command	Directory Handling	see B.1
	Bits 3–0 0001	Function	Change Current Directory	see B.2
Byte 2		TAN		see B.8
Bytes 3,4		Path Name Length		see B.12
Bytes 5–*n*		Path Name		see B.13

C.2.3.3 Change Current Directory Response

Transmission repetition rate:		In response to Change Current Directory Request message		
Data length:		8 bytes		
Parameter group number:		FS to client, destination-specific		
Byte 1	FS function = 17_{10}			
	Bits 7–4 0001	Command	Directory Handling	see B.1
	Bits 3–0 0001	Function	Change Current Directory	see B.2
Byte 2		TAN		see B.8
Byte 3		Error Code		see B.9
		0_{10}	Success	
		1_{10}	Access denied	
		2_{10}	Invalid Access	
		4_{10}	File, path or volume not found	
		7_{10}	Invalid destination name given	
		10_{10}	Media is not present [formerly the code for error 13, below, in Version 2 FSs (see B.5)]	
		13_{10}	Volume is possibly not initialized	
		43_{10}	Out of memory	
		44_{10}	Any other error	
Bytes 4–8		Reserved, transmit as FF_{16}		

C.3 File Access

C.3.1 Overview

File Access consists of messages for opening and closing files, for navigating through files and for reading and writing data to files.

C.3.2 Create File or Directory

Creating a file using the Create File command is accomplished by using the Open File command with the Flag set to create a new file.

C.3.3 Open File

C.3.3.1 General

Open File opens the file specified by the Path. Flags specify the mode in which the file is opened. On successful completion, the Open File Request returns the Handle to be used to identify the file in subsequent operations.

C.3.3.2 Open File Request

Transmission repetition rate:		On request		
Data length:		Variable		
Parameter group number:		Client to FS, destination-specific		
Byte 1	FS function = 32_{10}			
	Bits 7–4 0010	Command	File Access	see B.1
	Bits 3–0 0000	Function	Open File	see B.2
Byte 2		TAN		see B.8
Byte 3		Flags		see B.14
Bytes 4,5		Path Name Length		see B.12
Bytes 6–*n*		Volume, Path, File and Wildcard Name		see B.34

C.3.3.3 Open File Response

Transmission repetition rate:		In response to Open File Request message		
Data length:		8 bytes		
Parameter group number:		FS to client, destination-specific		
Byte 1	FS function = 32_{10}			
	Bits 7–4 0010	Command	File Access	see B.1
	Bits 3–0 0000	Function	Open File	see B.2
Byte 2		TAN		see B.8

Byte 3	Error Code		see B.9
	0_{10}	Success	
	1_{10}	Access denied	
	2_{10}	Invalid access	
	3_{10}	Too many files open	
	4_{10}	File, path or volume not found	
	6_{10}	Invalid given source name	
	8_{10}	Volume out of free space	
	10_{10}	Media is not present [formerly the code for error 13, below, in Version 2 FSs (see B.5)]	
	13_{10}	Volume is possibly not initialized	
	43_{10}	Out of memory	
	44_{10}	Any other error	
Byte 4	Handle		see B.10
Byte 5	Attributes		see B.15
Bytes 6–8	Reserved, transmit as FF_{16}		

C.3.4 Seek File

C.3.4.1 General

Seek File sets the file pointer for the next access within a file. Depending on the position mode, the file pointer can be set relative to the beginning of the file, the current file pointer or the end of the file. If the function succeeds, the new position is returned. The new file position is based on the following position mode:

0 New position = beginning of file + offset (can only be positive or 0 value);

1 New position = current position + offset (can be positive or negative value);

2 New position = end of file + offset (can only be negative or 0 value).

When the Handle references a file, the offset and position are defined in bytes. When the Handle references a directory, the offset and position are defined in Directory Entries.

When the file pointer is at the end of file position and a request to move the file pointer past the end of file is sent, the response shall contain a “File pointer at end of file” error code.

C.3.4.2 Seek File Request

Transmission repetition rate:		On request		
Data length:		8 bytes		
Parameter group number:		Client to FS, destination-specific		
Byte 1	FS function = 33_{10}			
	Bits 7–4 0010	Command	File Access	see B.1
	Bits 3–0 0001	Function	Seek File	see B.2
Byte 2		TAN		see B.8
Byte 3		Handle		see B.10

Byte 4	Position Mode	see B.17
Bytes 5–8	Offset	see B.18

C.3.4.3 Seek File Response

Transmission repetition rate:	In response to Seek File Request message			
Data length:	8 bytes			
Parameter group number:	FS to client, destination-specific			
Byte 1	FS function = 33_{10}			
	Bits 7–4 0010	Command	File Access	see B.1
	Bits 3–0 0001	Function	Seek File	see B.2
Byte 2		TAN		see B.8
Byte 3		Error Code		see B.9
		0_{10}	Success	
		5_{10}	Invalid Handle	
		11_{10}	Failure during a read operation	
		42_{10}	Invalid request length	
		43_{10}	Out of memory	
		44_{10}	Any other error	
		45_{10}	File pointer at end of file	
Byte 4		Reserved, transmit as FF_{16}		
Bytes 5–8		Position		see B.19

C.3.5 Read File

C.3.5.1 General

Read File reads data from the file referenced by the Handle. If the Handle refers to a file, Count specifies the number of data bytes to be read. The requested data (excluding the other parameters) is sent in the response (up to 1 780 bytes when TP is used, up to 65 530 bytes when ETP is used). The number of data bytes read can be less than requested if the end of the file is reached. If the Handle refers to a directory, Count specifies the number of directory entries to be read, while Report Hidden Files specifies whether files with attribute set to “hidden” are part of the directory entries listing.

When the file pointer is at the end of file position and a request to read past end of file is sent, the response shall contain a “File pointer at end of file” error code.

C.3.5.2 Read File Request

Transmission repetition rate:	On request			
Data length:	8 bytes			
Parameter group number:	Client to FS, destination-specific			
Byte 1	FS function = 34_{10}			
	Bits 7–4 0010	Command	File Access	see B.1
	Bits 3–0 0010	Function	Read File	see B.2

Byte 2	TAN	see B.8
Byte 3	Handle	see B.10
Bytes 4,5	Count	see B.20
Byte 6	Report Hidden Files	see B.28
Bytes 7,8	Reserved, transmit as FF_{16}	

C.3.5.3 Read File Response (Handle-referenced file)

The Read File Response message contains the data read from a file referred to by the Handle specified in the Read File Request message.

Transmission repetition rate:		In response to Read File Request message		
Data length:		Variable		
Parameter group number:		FS to client, destination-specific		
Byte 1	FS function = 34_{10}			
	Bits 7–4 0010	Command	File Access	see B.1
	Bits 3–0 0010	Function	Read File	see B.2
Byte 2		TAN		see B.8
Byte 3		Error Code		see B.9
		0_{10}	Success	
		1_{10}	Access denied	
		5_{10}	Invalid Handle	
		11_{10}	Failure during a read operation	
		42_{10}	Invalid request length	
		43_{10}	Out of memory	
		44_{10}	Any other error	
		45_{10}	File pointer at end of file	
Bytes 4,5		Count		see B.20
Bytes 6–n		Data		

C.3.5.4 Read Directory Response (Handle-referenced directory)

The Read Directory Response message contains the directory entries read from a directory, referred to by the Handle specified in the Read File Request message.

Transmission repetition rate:		In response to Read File Request message		
Data length:		Variable		
Parameter group number:		FS to client, destination-specific		
Byte 1	FS function = 34_{10}			
	Bits 7–4 0010	Command	File Access	see B.1
	Bits 3–0 0010	Function	Read File	see B.2
Byte 2		TAN		see B.8

Byte 3	Error Code		see B.9
	0_{10}	Success	
	5_{10}	Invalid Handle	
	11_{10}	Failure during a read operation	
	42_{10}	Invalid request length	
	43_{10}	Out of memory	
	44_{10}	Any other error	
	45_{10}	File pointer at end of file	
Bytes 4,5	Count		see B.20
Bytes 6–*n*	Directory Entries		see B.21

C.3.6 Write File

C.3.6.1 General

Write File writes data to an open file that is addressed by a Handle. The data (excluding the other parameters) to be written is sent to the FS in the request (up to 1 780 bytes when TP is used and up to 65 530 bytes when ETP is used). The Write File command shall not be used with a Handle that references a directory.

C.3.6.2 Write File Request

Transmission repetition rate:	On request			
Data length:	Variable			
Parameter group number:	Client to FS, destination-specific			
Byte 1	FS function = 35_{10}			
	Bits 7–4 0010	Command	File Access	see B.1
	Bits 3–0 0011	Function	Write File	see B.2
Byte 2		TAN		see B.8
Byte 3		Handle		see B.10
Bytes 4,5		Count		see B.20
Bytes 6–*n*		Data		

C.3.6.3 Write File Response

Transmission repetition rate:	In response to Write File Request message			
Data length:	8 bytes			
Parameter group number:	FS to client, destination-specific			
Byte 1	FS function = 35_{10}			
	Bits 7–4 0010	Command	File Access	see B.1
	Bits 3–0 0011	Function	Write File	see B.2
Byte 2		TAN		see B.8

Byte 3	Error Code		see B.9
	0_{10}	Success	
	1_{10}	Access denied	
	5_{10}	Invalid Handle	
	8_{10}	Volume out of free space	
	9_{10}	Failure during a write operation	
	42_{10}	Invalid request length	
	43_{10}	Out of memory	
	44_{10}	Any other error	
Bytes 4,5	Count		see B.20
Bytes 6–8	Reserved, transmit as FF_{16}		

C.3.7 Close File

Close File closes the file specified by the Handle. All internal buffers belonging to this file are written and the directory entry is updated. The Handle is invalid after the Close File Response message.

C.3.7.1 Close File Request

Transmission repetition rate:		On request		
Data length:		8 bytes		
Parameter group number:		Client to FS, destination-specific		
Byte 1	FS function = 36_{10}			
	Bits 7–4 0010	Command	File Access	see B.1
	Bits 3–0 0100	Function	Close File	see B.2
Byte 2		TAN		see B.8
Byte 3		Handle		see B.10
Bytes 4–8		Reserved, transmit as FF_{16}		

C.3.7.2 Close File Response

Transmission repetition rate:		In response to Close File Request message		
Data length:		8 bytes		
Parameter group number:		FS to client, destination-specific		
Byte 1	FS function = 36_{10}			
	Bits 7–4 0010	Command	File Access	see B.1
	Bits 3–0 0100	Function	Close File	see B.2
Byte 2		TAN		see B.8

Byte 3	Error Code		see B.9
	0_{10}	Success	
	1_{10}	Access denied	
	5_{10}	Invalid Handle	
	8_{10}	Volume out of free space	
	9_{10}	Failure during a write operation	
	43_{10}	Out of memory	
	44_{10}	Any other error	
Bytes 4–8	Reserved, transmit as FF_{16}		

C.4 File Handling

C.4.1 Overview

File Handling consists of messages used to move, copy and delete files, get or set attributes and get the date of a file.

C.4.2 Move File

C.4.2.1 General

Move File moves or copies a file from its current location to a new location. The type of action is specified by the File Handling Mode and destination specification:

a) if the destination filename differs from the file's present name, the file is renamed;

b) if the destination path differs from the source path, the file is moved;

c) if the destination path contains directories that do not exist, those directories are created;

d) if the “copy” mode is set, the file is copied.

If the directory or file exists in the destination path, an Error Code “Access denied” shall be returned unless the *force* mode is set. The *recursive* mode is required to move or copy a directory that contains further directories or files. If, in this case, the recursive mode is not set, an Error Code “Access denied” shall be returned. If the destination of a recursive move or copy is in the source path, then an Error Code “Access denied” shall be returned. When specifying a directory, add a “\” to the end to indicate in the Source or Destination Path Name. If the destination of a recursive move or copy is a subfolder of the source path, then an Error Code “Access denied” shall be returned.

C.4.2.2 Move File Request

Transmission repetition rate:	On request			
Data length:	Variable			
Parameter group number:	Client to FS, destination-specific			
Byte 1	FS function = 48_{10}			
	Bits 7–4 0011	Command	File Handling	see B.1
	Bits 3–0 0000	Function	Move File	see B.2
Byte 2	TAN			see B.8
Byte 3	File Handling Mode			see B.27