27 March 2007

Standard for Forest Data and communications

StanForD

SKOGFORSK

Contents

A. General

FILNAME

The filename consists of a base name and an extension separated by a period.

There are no StanForD limitations concerning permissible characters for filenames.

The extension name shall be three characters long while there is no limit concerning the length of the base name of StanForD-files.

FILE FORMAT

All data are stored and transferred in ASCII format. Variable 1 type 3 is used to specify the character set according to the ISO-standard, see the list of variables.

DATA

Data comprise variables and checksums (hash totals).

A variable consists of four components which, in the order they occur, are: Variable number, Type number, Data and End character. The components (and elements of the data) are separated by a delimiter.

The variable number and type number are numeric.

The data may be numeric or alphanumeric, as specified in the list of variables.

The data may comprise one or more elements, integers for numeric parts of the data and text strings for alphanumeric parts.

The characters used as delimiters (separators) are the space/blank (ASCII 32) and line feed, LF, (ASCII 10).

The following rules apply to delimiters:

- There must be at least one space (and no LF) between the variable number and the type.
- There must be at least one space between the type and the data field. If the data field is alphanumeric, it must start with a LF. This means that there must be both a space and a LF between a type and a alphanumeric data field.
- Text strings in an alphanumeric data field are separated by LFs.
- Integers in numeric data fields are separated by one or more spaces or LFs.

The end character is the tilde ~ (ASCII 126). This may be preceded by one or more spaces after numeric data fields.

All numeric data are decimal except when stated differently in the description of the variables (var151, var152, var141_t1, var276_t1 are always binary and var141_t3, var144_t2 may be binary).

Variables in a StanForD-file shall contain data, in other case they should not be included in the file.

When updating the standard aggregated or summed variables should not be added if these are possible to calculate based on variables already existing in the standard.

DESCRIPTION OF VARIABLES

Variables are defined by means of a variable number and a variable name. Only the variable number is sent during data transfer. In the case of vectors and multidimensional variables, the values used as the index are shown in the Description column. For example, 1...var111_t1 indicates that the index runs from 1 to the value of variable 111 type 1, NUMTREESPC, obtained earlier.

When several indexes are used, the data are stored and sent such that the first index specified will run fastest.

ABBREVIATIONS

The following abbreviations are used in all standard documents:

When writing an abbreviation for variable number, lowercase letters are used and, a space or underscore e.g. varible 35, type 2 is written var35 t2 or var35_t2

The abbreviation for file types are always written using lowercase letters, e.g. stm-file, apt-file

The name of a variable is always written using uppercase letters, e.g. CONTACTNO

DATA TYPES

The following data types are used:

String[80], long string[∞], integer[2 bytes] or long integer[4 bytes], see table 1.

In some cases, decimals are split between two variables, one for the integer part and one for the numerator (decimal places). The numerator part is assigned the variable number 1000 plus the variable number for the integer part. Only the three first decimals are registered (decimals/1000).

Table 1. Data types in StanForD

Data type	Range	Format
String	0 – 80 characters	0 – 80 char
Long string	0 – ∞ characters	0 – ∞ char
Integer	-32767 – 32767	2 bytes
Long integer	-2147483647 – 2147483647	4 bytes

Text strings in alphanumeric data fields may be up to 80 characters long (excluding the LF prefix).

SEQUENCE OF VARIABLES

Variables may occur in any order in the file except that those determining the size of others must precede those others.

DEPENDING VARIABLES

If variable A is used in a file and variable A is depending on variable B then variable B must be included in the file.

THE VALIDITY OF VARIABLES

If a variable is repeated in a file, the first value is valid until the new value has been read. Thereafter the new value is valid and the old one is not valid.

If a variable is repeated with another type, both the values are valid. For some variables one type can be dimensioned by an earlier type.

FILE TERMINATION

Every file is terminated by one or more checksums. Checksums are sent as variables without a type number. See variables 991, 992 and 993 in the list of variables.

The three checksums are computed from the beginning of the file up to and including the space after the variable number for checksum 1, i.e. 991. The method of calculation is described in Appendix 1. Each checksum is followed by a tilde ~ (ASCII 126).

Variable 991, checksum 1, is mandatory. If variable 992 and/or 993 is used, it must follow variable 991.

If a file has been edited or altered in any way following file transfer and the checksums have not been updated, the checksum variables must be removed.

SENDING STANDARD FILES AS ONE FILE

Combination files

The file-type suffix, cmb (combination file), shall be used when multiple standard files are assembled into one file. Each individual file is treated as a data block in the combination file.

Cmb	Block1	Block2	Block3	Block4

Example of CMB file containing four blocks

Variable 1, type 2, shall be used when a new data block (formerly a file) follows in the cmb file. The data are read one block at a time. If a variable is not repeated in the next block, its value in the last block in which it was given will be assumed.

Cmb	Prd	......	var111	......	prd	....	....	Stm	....	....

CMB <--------- Block1 PRD file ------> <--- Block2, PRD file---><--- Block3, STM file---->

Example of CMB file containing two PRD files and one STM file. N.B.: Var111 does not need to be repeated in Block2 if its value is the same as in Block1.

Compressing multiple files into a single file

Several files can be packed together into a single file with the use of a compression or ZIP program. Such programs also perform data compression, thus making the files smaller and the data transmission faster.

B. File and variable names reserved for specific applications

FILENAME EXTENSIONS

Extensions must be specific to the application concerned:

cmb:	Combination file. See above.
apt:	Bucking (cross-cutting) instructions including price matrixes per assortment.
prd:	Production (primarily harvesting production data).
pri:	Production-individual, harvesting data concerning each individual log and stem is registered.
prl:	Forwarder production data where data per load can be registered.
drf:	Operational monitoring data, covers both time (tid) and repair (rep) data since major update in 2003.
tid:	Time follow-up
rep:	Repairs follow-up
mas:	Machine variables
avs:	Instruction for bucking to taper
stm:	Stem values — measured length and diameter values
sti:	Stem ID. Sent from merchandising (bucking) computer to digital calipers or data logger to facilitate identification of control stems.
ktr:	Control measurements. Sent from digital calipers or data logger to (on-board) bucking (merchandising) computer, and from the bucking computer to an office system. Both operator, machine and auditor measurements of length and diameter can be included in the file.
kau:	Calibration criteria. N.B. Use of kau files is no longer recommended. Superseded by ktr files (as from 11 March 1996).
kal:	Calibration.
psu:	Summed production file.
hks:	Production variables for Germany.
inv:	Inventory variables.
oai:	The bucking identity of the harvesting site. The file contains data for a site which is a subset of the “apt-file”.
ghd:	Data describing harvesting object, primarily used by GIS applications.
spp:	Stem prediction parameters, parameters describing how to butt end diameters.
ap1:	Assortment specific data used together with oai-file in order to create an apt-file. Today only adapted for use in Finland. Some "machine specific" variables are not included in oai- or ap1-files as it is more efficient to set these in the machine. Ap1-files must be used together with an oai-file in order to create a complete apt-file.

Extensions reserved for individual matrices in APT and PRD files

The following *mandatory* extensions are used for individual matrices in apt and prd files:

apm:	Price matrix from apt files (part of variable 162)
fpm:	Requirements specification from apt files (part of variable 191)
prm:	Log tally from prd files (part of variable 201)

Old naming conventions no longer recommended (as from 21 October 1996)

anm:	old name for apm
fnm:	old name for fpm
pnm:	old name for prm

The naming conventions were changed owing to problems arising with files having a matrix number higher than 9 as the 'n' indicated the tree species number and the 'm' indicated the serial number of the price matrix for the given tree species.

Remaining

Machine variables, i.e. variables specific to a certain item of equipment, must be kept in a separate file. The filename must be unique and, ideally, indicative of the name of the equipment. Base names used must be submitted to Skogforsk. Files must be given the extension mas. Variables must be assigned a number within the range, 2000–2999. Variables are not standardised.

The following recommendations are made concerning the base name for stm files. The first four characters should be made up of optional text and the remaining characters should comprise a serial number. Examples of optional text are month and day or place name (e.g. mmdd0001.stm or ABCD0001.stm).

RESERVED VARIABLE NUMBERS

Variables 800–899 are reserved for use by the manufacturers during development and testing.

Variables 900-950 are reserved for use by users during development and testing.

Variables 1000–1999 are reserved for decimal numerators.

The following formats are available for older date variables:
Type 1: YYMMDD
Type 2: YYMMDDHHMM
Type 3: YYMMDDHHMMSS
Type 4: YYYYMMDDHHMMSS

New date variables are only implemented in the YYYYMMDDHHMMSS format.

C. Communications standard

This standard deals with the transfer of data to and from computers on board forestry machines. Using this standard, it should be possible for on-board computers, e.g. for bucking (cross-cutting), production reporting and the like, to communicate via a suitable interface with other computers or data recording devices.

The standard in no way precludes the on-board computers from communicating in other ways using suitable equipment, such as modems for radio communication.

The original draft of the standard was compiled by a group of experts in 1987 and was adopted in that year by a user group comprising representatives from Domänverket, Korsnäs AB, MoDo Skog AB, SCA Skog AB, Stora Skog AB and the Södra Skogsägarna (forest owner association).

COMMUNICATIONS PROTOCOL

The data-communications standard is essentially a mini version of the **Kermit** communications program. This version contains the commonest and most basic communications functions. Kermit should also be followed if users wish to incorporate additional functions.

A description of the Kermit protocol is contained in *Kermit User Guide, Sixth Edition, Revision 2, May 26 1986*. Up-to-date information can be found on the web-page of The Columbia University (http://www.columbia.edu/kermit/).

The version of Kermit used must contain the following basic functions:

- Transfer of ASCII files
- Terminal emulation
- Basic server functions
- Transmission of basic commands to the server
- Facility for interrupting file transfer

The standard shall include the following Kermit functions:
CONNECT
BYE
SEND
GET
SERVER
RECEIVE
REMOTE DIR
REMOTE DEL
REMOTE SPACE
QUIT
SHOW

The server function must support SEND, GET, BYE, REMOTE DIR, REMOTE DEL and REMOTE SPACE.

The functions are explained in the *Kermit Protocol Manual.*

Communication shall be over a half-duplex channel without a backward channel. The following parameters shall be used:

BAUD	1 200	
BLOCK-CHECK-TYPE	3	This also supports types 1 & 3 under Kermit
END-OF-LINE	CR	(ASCII 13)
EOF	NOCTRL-Z	

FLOW-CONTROL	None	
HANDSHAKE	None	
INCOMPLETE	Discard	
PARITY	None	
RECEIVE PACKET-LENGTH	94	
START-OF-PACKET	^A	(ASCII 1)
TIME-OUT (sek)	10	PCs & mainframes
	12	Data cassettes
	14	On-board computers
RETRY	10	
Stop bits	1	Stoppbit
Word length	8	Bits

CHANGING THE BAUD RATE

The standard specifies a baud rate of 1200. For applications such as updating price lists and accessing logging data, it may be desirable to set an optional baud rate for communication with a PC. This type of data transfer is usually achieved using an integrated software program such as SiliviA and WinApt.

If an alternative baud rate is chosen, it must be selected each time a transfer of data is to be made and reset to the default value (1200 baud) on completion.

Procedure

To change the Kermit baud rate in the on-board hardware (usually a data cassette), a "Remote host baud<baudrate>" command is sent by either the operator or the PC's communications program. As soon as an "Acknowledge" packet has been received, the baud rate is changed in both the on-board hardware and the PC. If the change cannot be made, an "Error" packet containing a text message is sent. The PC software that effects the baud-rate change—either on command or automatically—shall reset the baud rate to the default value when the operator exits the program.

If a "Break" signal is received by the on-board hardware or the PC, the baud rate must be set to 1200. If the previous rate was not the standard default value, a new "Break" signal must be sent. The reason for this is to provide additional security in case interference causes the hardware to recognize a spurious "Break" signal. Communication then proceeds as before. If a data packet has been disrupted by a "Break" signal or change of baud rate, the packet will be sent again as part of the normal Kermit procedure (such as if an incorrect checksum is noted) following a "Time-out".

A received "Break" signal means that all the bits in the received character, including any parity or stop bits, will have the value "0" (zero). This must apply at all baud rates. Thus, a "break" signal must have a duration of at least 275 ms.

An example of this procedure is given in chapter *"Example of how to change the baud rate"*

HARDWARE REQUIREMENTS

Connectors and pins

The interface for connection to the on-board computer must be a 25-pin D-type connector with sleeve, wired in conformance with CCITT V24, V28 and

ISO 2110. These same standards also specify the signal levels for the connector. The connections must be those shown in the table below.

The fact that the connector in the machine (DTE) has a sleeve is a conscious departure from the standard for safety reasons. The possible use of pin 9 for power supply to the connected hardware constitutes an addition to the standard.

Connection to the on-board hardware can be made either via a connector on the computer or by means of an adaptor supplied by the manufacturer.

D connector conforming CCITT stift nr	**Signal #**	**Signal description**
1	101	Protective ground/Cable screen
2	103	Transmitted data
3	104	Received data
4	105	Request to send
5	106	Ready for sending/Clear to send
6	107	Data set ready
7	102	Signal ground/Common return
8	109	Data channel received line signal detector (carrier detect)
9		+(10–30 V) supply
20	108	Data terminal ready
22	125	Calling indicator/Ring indicator

Signal levels

All signalling is effected by a High or Low signal level being sent. During transmission, a High signal level is represented by a voltage above + 5 V and a Low signal level by a voltage below - 5 V. During reception, a signal strength higher than + 3 V is interpreted as a High signal level and a signal below - 3 V as a Low signal level. The nominal level for transmission should be ± 12 V.

Signal logic

It should be noted that transmission over the two data lines has negative logic, i.e. the binary digit 1 is represented by a negative (Low) signal and a binary 0 by a positive (High) signal. Between each transmitted character, the signal level is Low, i.e. a logical 1. Every character is preceded by a start bit, which is represented by a High signal or logical 0. Stop bits are represented by a logical 1 (Low), which is also the same as the "awaiting signal" level, signifying the minimum delay time between characters being transmitted.

All other signals have positive logic.

Supply voltage

The hardware to which a data cassette is to be connected must provide through pin 9 a positive feed, relative to pin 7, of 10–30 V to the data cassette. Its drive capacity should be 0.3 A and limited to a current of 0.5 A.

EXAMPLE OF HOW TO CHANGE THE BAUD RATE

The small print following the Rpack examples denotes an optional text string.

1. To change the baud rate from 1200 to 19200

```
<1200 baud>
Spack: ^A0 I~- @-#Y1~* ~G^M
Rpack: ^A0 Yz* @-#Y1~" zD^M
Spack: ^A- CBAUD 19200+^M
Rpack: ^A7 YNew baud rate: 19200Z^M
<This changes the baud rate to 19200>
```

2. Attempt to change the baud rate to an invalid value

```
<1200 baud>
Spack: ^A0 I~- @-#Y1~* ~G^M
Rpack: ^A0 Yz* @-#Y1~" zD^M
Spack: ^A- CBAUD 12345.^M
Rpack: ^AH EVMU-512 error 133, Invalid baud rate!^M
<1200 baud>
```

3. Change of baud rate with "Break" to 1200 baud.

```
<1200 baud>
Spack: ^A0 I~- @-#Y1~* ~G^M
Rpack: <Invalid or absent message>
<Send "Break" and wait at least one "time-out">
Spack: ^A0 I~- @-#Y1~* ~G^M
Rpack: ^A0 Yz* @-#Y1~" zD^M
```

D. Kermit checksums

CHECKSUM TYPE 991

'S' is computed as the arithmetic sum of all ASCII codes up to and including the space character after the variable number 991.

'S' is then converted using the following formula:

S: = (S + ((S AND 192) DIV 64)) AND 63
[S: = (S +((S AND 0C0h) SHR 6)) AND 03Fh]

'S' is written to the file as a numeric variable in ASCII format and is terminated with a tilde (ASCII 126). (Note: The type should not be included.)

CHECKSUM TYPE 992

'S' is computed as the arithmetic sum of all ASCII codes up to and including the space character after the variable number 991.

'S' is then converted using the following formula:

S: = S AND 4095 [S: = S AND = 0FFFh]

'S' is written to the file after variable 991 as numeric variable 992 in ASCII format and is terminated with a tilde (ASCII 126). (Note: The type should not be included.)

CHECKSUM TYPE 993

'S' is computed by a 16-bit cyclic redundancy check (CRC) on all ASCII codes up to and including the space code after the variable number 991. (See description in Kermit User Guide.)

'S' is written to the file after variable 991 (or, possibly, after 992) as INTEGER variable 993 in ASCII format and is terminated with a tilde (ASCII 126). (Note: The type should not be included.)

Checksum 993 is interpreted as a 16-bit integer when it is to be written in ASCII format in the file.

E. Listing of variables

All standardized variables are described in separate documents. These documents are available at www.skogforsk.se.