The Software-independent Data Format

Background

The use of data 1ogging equipment to capture data is now well established in school science, with a prominent place in the UK National Curriculum. There is a growing realistic in that captured data can be further investigated with the aid of the powerful data-handling software now widely available in schools. Traditionally, data-capture software developed for schools did not support data export, so the data could only be explored within the original data-capture program. When developers, within the forum provided by the Educational Data Monitoring and Control Group began to discuss resources to facilitate the transfer of data between software packages, the issue of file format for data was closely examined. CSV is used widely in data-handling software, but was perceived to present two obvious difficulties-.

i it is an extremely loose specification with many fundamental variations. (In the last year the File Formats Working Group of Acorn Computers Ltd, Cambridge, England, has attempted to tighten the definition of CSV for Acorn RISC OS computers, but on other platforms the definition remains unclear.)

ii CSV files only contain data, with little supporting information about the origin or type of data, The most that can be offered is field names. Thus it is difficult for software to handle the data intelligently, and data files stored on disk contain no easily read information to indicate what the data refers to, a serious issue when managing data storage.

In order to solve both these difficulties, across all computer hardware platforms, the Software-Independent Data (SID) format was specified by the Educational Data Monitoring and Control Group (EDM&CG) and published in the first National Council for Educational Technology Technical Bulletin - Resources for data monitoring and control for science education, in 1991. The SID file format offers a common clearing house for data moving between different software applications. It has been adopted by sixteen products at the time of writing (see Appendix 3 for details), some of which are entirely new, and some of which have been revised to include SID support. The range includes both data-capture and data-handling packages. These packages may create and/or read SID files. Some use SID as their internal file format, others use simple utilities which change the proprietary file format to or from SID. In this way, any developer creating a software application to capture or handle data can ensure that data files can be imported and exported between the new and existing packages.

The status of SID

Prior to publication in 1991, a description of the agreed format was circulated for comment to group members and other interested parties (Appendix 2 lists all other parties who have been invited to comment on the SID format). That consensus was published as a proposed specification by NCET in a discussion paper, entitled 'Software Independent Data format - a file format for school science which facilitates data transfer between data-capture and data-handling software'. A small number of minor, but significant, changes were made subsequently in response to comment from a wider audience. The final specification was published in the 1991 NCET Technical Bulletin - Resources for data monitoring and control for science education. Since that point the SID specification remains unchanged and should be regarded as stable.

During the eighteen months since publication, there has been considerable feedback on the use of SID. In order to assist developers, clarification of some points is included here, along with recommendations for good practice which have evolved during use. These do not represent changes to the SID specification; they merely expand the explanations given previously.

Overview

The file format is known as the Software-independent Data format, SID. The MS-DOS file extension is SID. The file type is registered with Acorn Computers Limited as &C7D, name SID. The RISC OS icon for an SID file is a square with the letters horizontally across it. The definition of the icon is precise and must be adhered to; a pixel-fcr-pixel diagram is given in Appendix 4.

An SID file is an ASCII (text) file that consists of a header followed by a data file. The data format is based on values separated by commas, with the addition of a header to allow for transfer of additional information.

The header is compulsory for an SID file. It consists of a list of 'commands', each command starting with two per cent symbols to identify it. The minimum requirement is two commands: the identifier, which must be the first command and the data size, which will be the second command. All commands used must be listed, in lines, preferably in the order given below, before the data block. Each command occupies one line terminated by a carriage return and line feed. Software reading the file should ignore any command it does not recognise and have sensible defaults such that all commands, other than the identifier and data size, can be considered optional. It is essential that software does not crash if an SID file header contains more information than can be used by the importing package. Everything following the header is data.

SID is totally case independent, i.e. users should not assume case dependence at any point.

When writing, a carriage return (CR) followed by a line feed (LF) must be used to terminate all lines. The CR and LF therefore acts as command line separator in the header and record separator in the data block.

The Software-independent Data Format

Parameters will be separated with a comma. It is also possible that some data fields may be blank and thus will be expressed as two commas without an item of data between them.

NB The minimum specification for an SID file is a two command header followed by data as values separated by commas. To be SID-compatible, any software reading an SID file should accept such a file without hanging, and treat any other commands as optional, using them if present but having sensible defaults where they are absent.

Use of ASCII characters

An SID file contains only ASCII characters from the decimal set 10 (LF),13 (CR), and 32 to 126 inclusive. NB This does not include ASCII 9 (TAB) or ASCII 127 (delete or backspace). It should be noted that the following

ASCII characters have special significance:
A comma, ASCII 44, can only be used to separate fields.
White space, ASCII 32, is transparent and used only for clarification. See also 'Expressing numbers'.
%% as the first two non-white space characters of any line denote a command.
Note that the symbols < and > are not part of the command, but are shown in this document to clarify layout.

Other non-alphanumeric characters, e.g. single or double quotes, are textual characters with no special meaning and can only be used in strings.

The Software-independent Data Format

Writing data

There are only two data types - number and string. It is assumed that data is numeric unless otherwise stated, hence the provision of the special field unit string in the SID specification. Strings must be identified using the special field unit string. Thus data files including data as strings must include in the header the line:
%%fieldunit, fieldnumber, string CR LF
This allows data filters to screen out string data in packages which cannot handle them.

Data values are separated by commas. Some data fields may be blank, so will be expressed as two commas without an item of data between them.

Each record occupies a new line and lines are terminated using a carriage return (CR) followed by a line feed (LF).

It is normal practice for a field in a record, in any data tile format, to be of one type (or a union of types). An SID file can also, in line with normal practice, only contain one record structure.
E.g. the data block:

6,car,7,12 CR LF
7,van,5,8 CR LF VALID DATA
5,moped,8,12 CR LF

where field2 is declared as a string in the file header is within the SID format, as it has one consistent record structure. E.g. the data block:

car,van,moped CR LF 6,7,12 CR LF
7,5,8 CR LF INVALID DATA 5,8,12 CR LF

is not within the SID format as the record structure is inconsistent.

Expressing numbers

Numbers are expressed as decimal values. The characters +,-,.,0,l, 2,3,4,5,6,7,8,9 are accepted. The use of the + is optional, since numbers are assumed positive unless otherwise stated.
Exponentials are not supported as they cannot be expressed using the above terms.

Non-integer numbers must be expressed using a decimal point where decimal fractions are present; integers may make optional use of the decimal point, e.g. 1.0, 1.00, 1. and 1 should all be recognised as identical. Number fields may be blank where no value is recorded.

No limit is declared for the length of a numeric data value in the SID specification. Users are advised that when writing or reading any numeric value or string it is impractical to allow more than 255 characters.

The Software-independent Data Format

The structure of an SID file

Part 1 Standard commands for inclusion in the header

Note that the symbols < and > are not part of the command, but are shown in this document to clarify layout.

Commands used should be listed in the order shown.

%%Identifier, <file type>
This *must* be the first command, used to identify the file/header. The file type string is SID.

NB The whole file, including file type is case independent, i.e. SID, sid and Sid should all be accepted.

%%datasize, <number of records>,<number of fields>
This is the other compulsory command.

NB It is recommended that commands be listed in the order shown, making this the second command. The data file may not contain any records, but have a field structure, therefore datasize values of 0,0 or 0,n (where n represents any positive integer greater than 0) may occur.

%%title, <title>
This title will generally appear at the top of any display or window. The maximum length is 30 characters.

%%filedescription, <description>
The description is that of the data contained in this file, and is not formatted in any way. Some software may need to truncate this to 200 characters, so this is the maximum length.

%%fieldname, <field number>, <field name>
Where the field number is an integer from 1 upwards, there will be no theoretical limit to the number of fields (or channels in data-logging terms). The field name is usually used to describe the sensor or the parameter measured. It should be limited to a maximum of 15 characters.

%%fieldunits, <field number>, <field units>
Field units are a string: software reading this string will normally display it with the field name. Software writing this string will usually use standard units of measurement. (See Special field units below.) However, they will have to be exactly specified by the developer if software is to recognise them and allow conversion to other suitable units. Non-ASCII symbols, such as that normally used for degrees in temperature measurement, will have to be avoided and written in full. This string has a maximum of fifteen characters.

The Software-independent Data Format

%%fielddescription, <field number>, <field description> The description will be limited to 200 characters.

%%starttime, <HHMMSS>

The start time of the data capture is expressed as a six-digit number. HH is for hours, MM for minutes and SS for seconds, e.g. 145707 for 2.57 and 7 seconds pm.

%%stoptime, <HHMMSS>
The stop time of the data capture is expressed as a six-digit number. HH is for hours, MM for minutes and SS for seconds, e.g. 025807 for 2.58 and 7 seconds am.

%%startdate, <YYMMDD>
The start date of the data capture is expressed as a six-digit number. YY is for the year, MM the month and DD the day, e.g. 921204 for 4th December 1992.

%%stopdate, <YYMMDD>
The stop date of the data capture is expressed as a six-digit number. YY is for the year, MM the month and DD the day, e.g. 031206 for 6th December 2003 (it is assumed that no data will predate 1950).

%%interval, <seconds>
The interval between readings may be specified. The presence of an interval command represents a compressed field of time data. An interval of 0 may be used to signify time-independent data. If channels are sampled at different rates, then the minimum interval should be listed; a record sampled at a larger interval may therefore not contain data for each field. In such cases an unused field will be blank, i.e. will appear as two commas without an item between them. Intervals of less than one second will be expressed as a decimal, e.g. 10 milliseconds = 0.01.

%%maxmin,<fleld number>,<maximum value>,<minimum value>
maxmin refers to the maximum and minimum useful values in a data set.

NB The actual data may fall within this range or outside it, e.g. a thermistor may give readings in response to a temperature range of -20 to 150 degrees Celsius, but the readings may only be reliable in the range 0-100. In this case the maxmin command might read:
%%minmax, 1,+100.0,0 CR LF.

However, the actual data values read might fall in the range 25-100. The minmax command is useful for filtering out readings outside the reliable range, and for setting the limits of a graph axis, for example.

%%comment, <string>
Comments may be added as required.

The Software-independent Data Format

Special field units

Developers are encouraged to use standardised field units, enabling software to recognise appropriate ones. These must be expressed using standard ASCII characters. Where possible these units will be given as normal SI units, but attention is drawn to the fact that the SID format simply specifies a string. Defining standard units clearly lies beyond the scope of this document but developers' attention is drawn to the Association for Science document SI Units, Signs, Symbols and Abbreviations for guidance in the use of appropriate units.

Examples of what might be found in the field unit string are given below:

Parameter	SI unit	Alternative string
Time	s	second(s)
Length	m	metre(s)
Frequency	Hz	hertz

Special cases of non-SI units have been identified; these are listed below:

Specials:

time	Indicates that the field contains data in the form HHMMSS
date	Indicates that the field contains data in the form YYMMDD
mark	Indicates that the field contains only data marks of value 0 or 1 (1=marker) ASCII 48 (a space or 'no mark') or 49 (a mark)
string	Indicates that the field is data which has no numeric interpretation

Part 2 The data which follows the header

Record 1 field 1, Record 1 field 2, …., Record 1 field N CR LF
Record 2 field 1, Record 2 field 2, …., Record 2 field N CR LF

Numeric values will be stored at the maximum available resolution. Data handling software reading this data may limit the resolution used.

The Software-independent Data Format

New optional commands

NB The minimum specification for an SID file is a two-command header followed by data as values separated by commas. To be SID-compatible, any software reading an SID file should accept such a file without hanging, and treat any other commands as optional, using them if present but having sensible defaults where they are absent.

The SID specification allows for the introduction of new optional commands. One new optional command is now in use:

%%maxmin,<field number>,<maximum value>,<minimum value>
maxmin refers to the maximum and minimum useful values in a data set.

NB The actual data may fall within this range or outside it, e.g. a thermistor may give readings in response to a temperature range of -20 to 150 degrees Celsius. However, the readings may only be reliable in the range 0100. In this case, the maxmin command might read: %%maxmin,1,+100.0,0 CR LF

However, the actual data values read might fall in the range 25-1 00. The maxmin command is useful for filtering out readings outside the reliable range, and for setting the limits of a graph axis, for example.

Proprietary command prefixes
A new development in the use of the SID file format is the use of proprietary command prefixes. This has arisen through the extension of the use of SID for proprietary data files, not merely as a node for the exchange of data files. This extended use is extremely welcome, but to avoid the possibility of two new commands being introduced with the same name but different meanings, proprietary command prefixes have been established and must be used before any new command.

Proprietary command prefixes should be a minimum of two characters, preceded by %% and ended with an underscore, e.g. the new LogiT command 'sensor' would be written thus:
%%LogIT_sensor,1,14 CR LF
and the new Educational Electronics command 'sensorname' would be shown thus:
%%EE_sensorname,Light CR LF

ALL proprietary command prefixes MUST be registered with the Convenor of the EDM&CG (NO LONGER POSSIBLE BUT YOU GET THE IDEA) Those registered so far are:

Prefix: %%DH_
Owner: Data Harvest Group Ltd
Contact: Mr S D Allen at Educational Electronics
Use: Future Data Harvest products

Prefix: %%NCET_
Owner: National Council for Educational Technology
Contact: Keith Hemsley, NCET
Use: Future software

The Software-independent Data Format

Prefix: %%EE_
Owner: Educational Electronics
Contact: Mr S D Allen at Educational Electronics
Use: Future Educational Electronics products

Prefix: %%PHE_
Owner: Philip Harris Education
Contact: John Crellin at Philip Harris
Use: DL Plus and future products

Prefix: %%SCC_
Owner: SCC Research
Contact: Steve Cousins at SCC Research
Use: SCC products

Prefix: %%LogIT_
Owner: LogiT Project
Contact: Steve Cousins at SCC Research
Use: LogIT specific features

Prefix: %%GBX_
Owner: Minerva Software
Contact: Merlyn Kline at Minerva
Use: Graph Box Professional

Prefix: %%HC_
Owner: Homerton College IT Unit
Contact: Angela McFarlane at Homerton College
Use: Software products

When a proprietary command prefix is used in an SID header, it is recommended that a comment relating to ownership is also included. Developers should not adopt another user's proprietary commands without close liaison with the owner.

This development of the use of SID does not weaken the SID format. **It has always been the case that software reading SID must be able to ignore any commands it cannot use.** The extensions allow developers to use one file format for data that can be used to convey all the information needed in proprietary software, and from which data and a chosen subset of header information can be extracted reliably by other software.

Comments on the presentation of time-related data

Much debate has taken place over the presentation and use of data relating to elapsed time between data readings. The SID specification is quite clear in what it allows:

There are two ways of including this data in an SID file, by use of the %%interval command (which effectively represents a compressed field), and/or as a field of individual time data.

The Software-independent Data Format

Any software reading an SID file must be prepared to deal with either method of presenting this data where present. What follows is simply a consideration of related issues which may be helpful when deciding how to implement the use of SID files.

The debate has focused on the issue of selecting data to be used for plotting the x co-ordinates on a graph. This is particularly pertinent to much of the likely use of SID files.

If data is time dependent, time is likely to be the variable used on the x axis. Many applications creating data arrays of time dependent data show elapsed time as field 1. However, the data may not be time dependent, time may not be field 1, and/or the user may wish to plot against an alternative field. If a software application importing SID files to be displayed on a graph is to perform predictably without limiting user options unnecessarily there are several options open to the programmer; some of these are outlined below:

i Where no %%interval command is included in the header, data may be plotted automatically against field 1 (often time), or according to the record number.

ii Where %%interval is included in the header this may be used to generate the x co-ordinates automatically. Programmers must remember that if elapsed time is also included as a field they will generate a line on the graph of elapsed time v interval time. If the field containing the time data carries a field name including the word time, it could be recognised before plotting.

iii The selection of data used to plot the graph can be left to the user. The data file could be interrogated by the software to determine the possible options which are then made available for user designation as x or y co-ordinates.

When writing SID files it has become general practice to use %%interval or include time elapsed data as a field (usually field 1). Programmers may wish to take this into account when deciding on defaults for reading or writing SID files.

Summary

1 An SID file must begin with a header consisting of a minimum of two command lines: %%identifier and %%datasize. All other commands are optional but should be used in the order shown in the following section. New commands must have a registered proprietary prefix.

2 When writing an SID file a carriage return (CR) ASCII 13 and line feed (LF) ASCII 1 0 must be used to terminate every command line and every record in a file.

3 A line in an SID file may be:

i a command line in the header
ii a record in the data section
iii a blank line containing only CR LF (e.g. where no data exists, or if a reading has been missed in a single field record where elapsed time is described using the interval command).

Lines are made up of two types of string elements: textual string elements, or numerical string elements.

A textual string element may contain a mixture of ASCII characters ASCII 33 to 43 and ASCII 45 to ASCII 126 inclusive. A textual string element must not contain ASCII 13 (CR), ASCII 10 (LF) or ASCII 44 (comma).

The white space character, ASCII 32, is used to separate string elements for clarity, e.g. to separate words in a comment string.
Only command line strings can have the symbols c'/'O% as the first characters in the line.

A numerical string element is used to represent a numerical value in a header command line or as an item of data in a record.

A numerical string element may contain only the ASCII characters ASCII43, 45, 46 and 48 to 57.

NB This excludes ASCII 32, the white space character. If ASCII 43 or 45 (+ or -) is included it must only appear once at the beginning of the string. ASCII 43 and 45 are mutually exclusive.

The period character, . , ASCII 46, must be included when the numerical value represented by the string has a decimal fraction as part of its value.

Comma, ASCII 44, can be used ONLY to separate parameters in a command line or fields in a record. In either case the presence of white space characters before and/or after the comma will have no special significance, e.g.

%%identifier,sid CR LF is the same as %%identifier, sid CR LF

Example SID files

White space is used for visual clarity; the positions of carriage return (CR) and line feed (LF) are shown.

Using a full header with no proprietary commands

%%identifier, SID CR LF
%%datasize, 6,3 CR LF
%%title, pH and Temperature CR LF
%%filedescription, Jo Bloggs pH and temperature experiment 1st Oct 90 CR LF
%%fieldname, 1, Time CR LF
%%fieldunits, 1, Seconds CR LF
%%fielddescription, 1, Time from start (readings every 10 seconds) CR LF
%%fieldname, 2, pH CR LF
%%fieldunits, 2, CR LF
%%fielddescription, 2, Standard glass pH probe CR LF
%%fieldname, 3, Temperature CR LF
%%fieldunits, 3, degrees C CR LF
%%fielddescription, 3, Chemical resistant temperature sensor CR LF
%%interval, 10 CR LF 01
%%starttime, 1 53000 CR LF
%%stoptime, 153120 CR LF
%%startdate, 901001 CR LF
%%stopdate, 901001 CR LF
%%maxmin,3,-100.0,-10.0 CR LF
0,7,25.6 CR LF
9.9,7,25.6 CR LF
19.8,7.1,25.7 CR LF
29.7,7.6,25.1 CR LF
39.9,7.5,25.0 CR LF
49.9,7.4,24.9 CR LF

Using the minimum permitted header

%%identifier, sid CR LF
%%datasize,9,3 CR LF
0,7,25.6 CR LF
10,7,25.6 CR LF
20,7.1,25.7 CR LF
30,7.6,25.1 CR LF
40,7.5,25.0 CR LF
50,7.4,24.9 CR LF
60,7.4,25.0 CR LF
70,7.3,25.3 CR LF
80,7.3,25.4 CR LF

Using a partial header, showing data channels sampled at differing rates

%%identifier, SID CR LF
%%datasize, 9,3 CR LF
%%title, pH and Temperature CR LF
%%fieldname, 1, Time CR LF
%%fieldunits, 1, Seconds CR LF
%%fielddescripticn, 1, Time from start at 10 second intervals CR LF
%%fieldname, 2, pH CR LF
%%fieldunits, 2, CR LF
%%fielddescription, 2, Standard glass pH probe (readings every 20 seconds) CR LF
%%fieldname, 3, Temperature CR LF
%%fieldunits, 3, degrees C CR LF
%%fielddescription, 3, Chemical resistant temp. sensor (readings every 10 seconds) CR LF
%%interval, 10 CR LF
0,7,25.6 CR LF
10,,25.6 CR LF
20,7.1,25.7 CR LF
30,,25.1 CR LF
40,7.5,25.0 CR LF
50,,24.9 CR LF
60,7.4,25.0 CR LF
70,,25.3 CR LF
80,7.3,25.4 CR LF

Document scanned RF July 1999 – phantom spaces may have appeared etc.