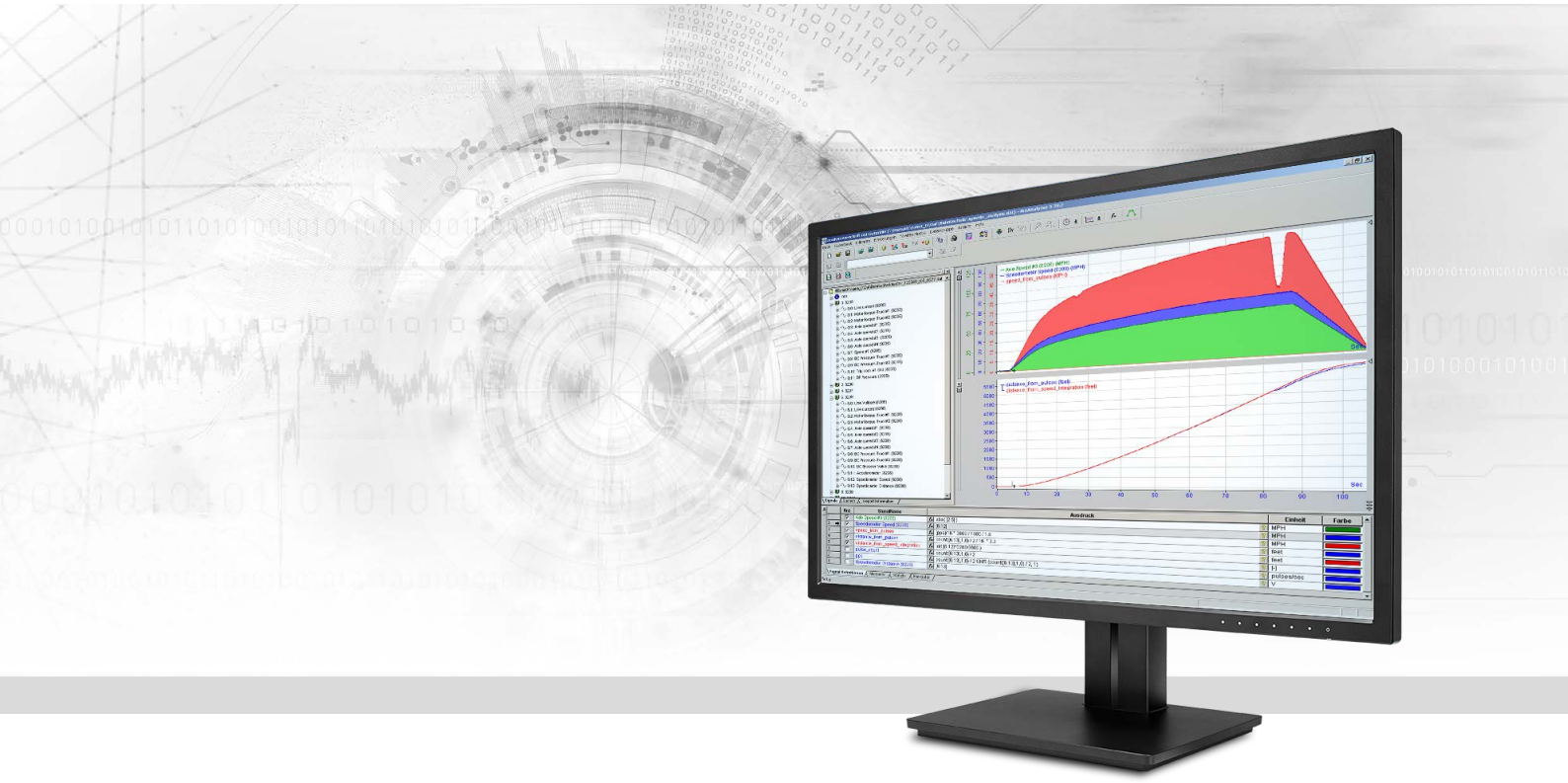


ibaAnalyzer-E-Dat

Import Interface for External Data File Formats

Manual

Issue 2.2

Measurement Systems for Industry and Energy

www.iba-ag.com

Manufacturer

iba AG

Koenigswarterstrasse 44

90762 Fuerth

Germany

Contacts

Main office	+49 911 97282-0
Support	+49 911 97282-14
Engineering	+49 911 97282-13
E-mail	iba@iba-ag.com
Web	www.iba-ag.com

Version	Date	Revision	Author	Version SW
2.2	03-2024	Expansion of the supported formats	RM	8.0.0

Contents

1 About this documentation

This documentation describes the function and application of the software

ibaAnalyzer-E-Dat.

1.1 Target group and previous knowledge

This documentation is aimed at qualified professionals who are familiar with handling electrical and electronic modules as well as communication and measurement technology. A person is regarded as professional if he/she is capable of assessing safety and recognizing possible consequences and risks on the basis of his/her specialist training, knowledge and experience and knowledge of the standard regulations.

1.2 Notations

In this manual, the following notations are used:

Action	Notation
Menu command	Menu *Logic diagram*
Calling the menu command	*Step 1 – Step 2 – Step 3 – Step x* Example: Select the menu *Logic diagram – Add – New function block*.
Keys	<Key name> Example: <Alt>; <F1>
Press the keys simultaneously	<Key name> + <Key name> Example: <Alt> + <Ctrl>
Buttons	<Key name> Example: <OK>; <Cancel>
Filenames, paths	`Filename`, `Path` Example: `Test.docx`

1.3 Used symbols

If safety instructions or other notes are used in this manual, they mean:

Danger!

The non-observance of this safety information may result in an imminent risk of death or severe injury:

- Observe the specified measures.

Warning!

The non-observance of this safety information may result in a potential risk of death or severe injury!

- Observe the specified measures.

Caution!

The non-observance of this safety information may result in a potential risk of injury or material damage!

- Observe the specified measures

Note

A note specifies special requirements or actions to be observed.

Tip

Tip or example as a helpful note or insider tip to make the work a little bit easier.

Other documentation

Reference to additional documentation or further reading.

2 ibaAnalyzer import interface

2.1 Supported file formats

Usually, data files can only be opened and analyzed with *ibaAnalyzer*, if they have iba's proprietary format and the file extension *.dat*.

With the *ibaAnalyzer-E-Dat* additional license which has to be enabled in the dongle, more file formats can be read.

ibaAnalyzer supports the import and the analysis for the following file formats, provided there is an active *ibaAnalyzer-E-Dat* license.

- ASCII (.txt, .csv)
- COMTRADE CFF (*.cff)
- NI TDMS (.tdm, .tdms)[1]
- Vista Control (*.varc)
- Apache Parquet
- FDA (*.das)
- Matlab (.mat)
- Wave (*.wav)
- PQDIF (*.pqd)
- Universal 58 (*.unv, *.bunf, *.uff)
- Vold files (*.tra, *.tor, *.spl)

[1] For more information about processing TDMS files, please consider the separate notes in the chapter *TDMS files*, Page ↗ *TDMS files*, page 21.

Reference

For detailed information about the general functions of *ibaAnalyzer*, please see the *ibaAnalyzer* manual.

The "Open data file" dialog only shows files with the extensions *.dat*, *.hdq*, *.pdc*, *.zip*, *.rar* etc. as well as *.txt*, *.csv*, *.tdm, .tdms, .das, .parquet* or *.mat*. Separate filters are availaible for the corresponding extensions. Depending on the file type, different settings are available in order to read the data.

2.2 ASCII files

These files can be created using different tools, e.g. a text editor, MS Excel or other programs which can export data as text file.

For *ibaAnalyzer* being able to import the files and display the data contained properly, the user has to observe some general conditions when formatting the ASCII files. Nevertheless, when configuring the import interface, different options provide for a certain flexibility.

The dialog for configuring the import interface opens automatically, as soon as you open an ASCII file via the "Open Data File" dialog or drag it from the Windows Explorer via Drag & Drop to the *ibaAnalyzer* signal tree window.

Tip

In case you want to import ASCII measurement files in *ibaAnalyzer*, you should assign the endings .txt or .csv. Then, the files can be displayed in the browser of the "Open Data File" dialog.

Please do not assign the .dat ending, as *ibaAnalyzer* expects and accepts only original iba measurement files with this ending.

Other file endings than the ones mentioned above, are not accepted.

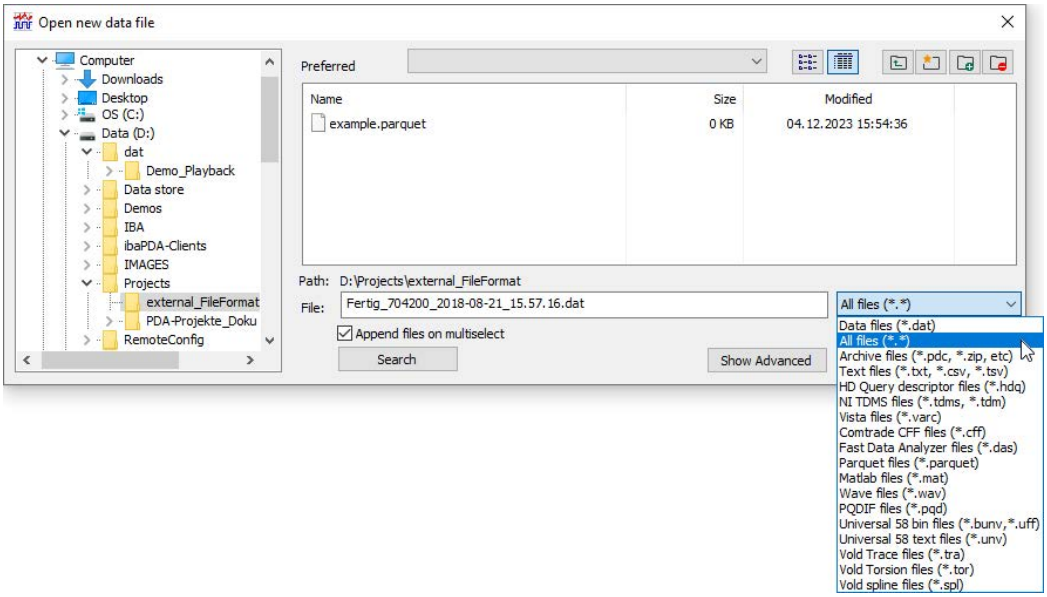
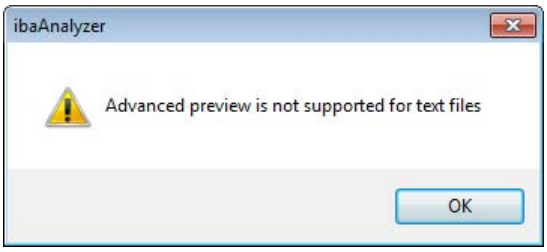


Fig. 1: Dialog for opening measurement files

Note

The advanced file preview in the “Open data file” dialog is not supported for most of the external formats. You will be notified when you open a text file while “Show advanced” is selected.



Acknowledge the message by clicking <OK> and continue to open the file. You can avoid the message by disabling the advanced preview.

If you select a file with the .txt or .csv ending, and click on <OK>, the dialog for configuring the import interface will be opened:

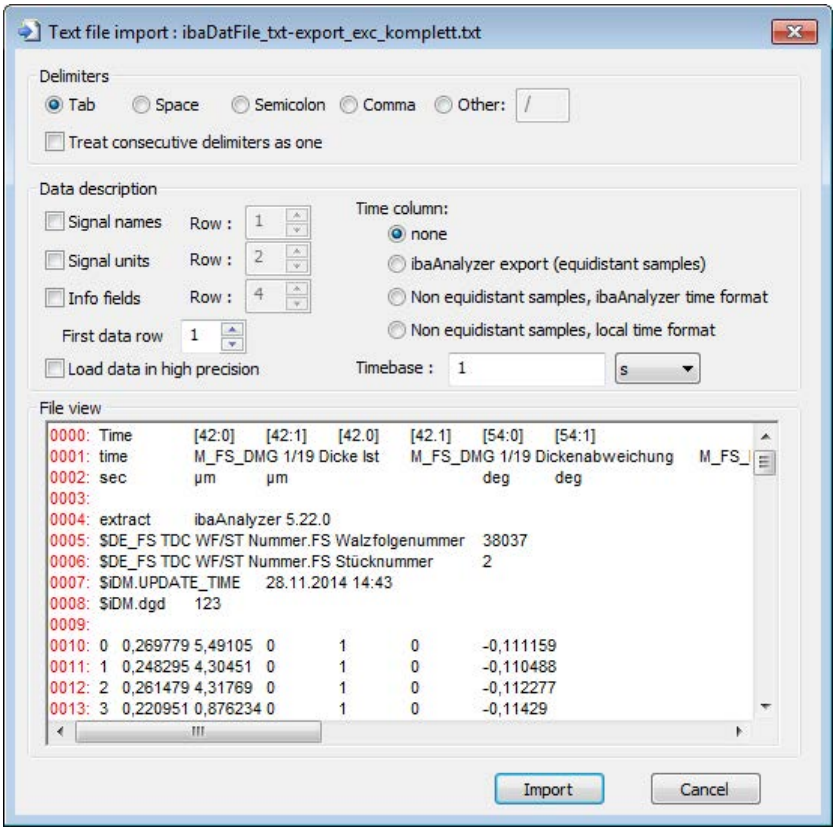


Fig. 2: Dialog for importing measurement files in ASCII format

In this dialog, you can do the settings according to the format of the source file.

The presets are taken from the settings of the recent successful import in the current or in a previous *ibaAnalyzer* session.

Delimiters

Tab, Space, Semicolon,...
Please choose the delimiter which separates in the source file parameters and series of measurement values. When exporting into an ASCII file, *ibaAnalyzer* uses by default the Tab for separation and hence rather generates TSV files (tab separated values) than CSV files (comma separated values). For exporting as well as for importing, you can also use real CSV files with a comma as delimiter. This only works if for the (analog) measurement values not a comma is used as decimal separator. Other delimiters like space, semicolon or user defined signs are also accepted. It goes without saying, that in the case of spaces as separator neither signal names nor Info fields or Technostrings may contain spaces.

Treat consecutive delimiters as one
If this option is disabled (Default) and in the source file there are two or more consecutive delimiters (e.g. "Value1;Value2;;Value4;…"), then empty signals will be created in the *ibaAnalyzer* signal tree.

In case this option is enabled, the consecutive delimiters are treated as one, so that no empty signals or gaps are created.

Tip

Activating this option might be useful, when data from a spreadsheet program, e.g. MS Excel are imported into a CSV file for analyzing it with *ibaAnalyzer*. For programs like these, it is customary that empty columns are inserted, e.g. for creative reasons. Empty columns generally create consecutive delimiters in the ASCII export.

2.2.1 Generating an appropriate ASCII file

In the preceding chapter, we have described the essential parameters *ibaAnalyzer* needs for reading ASCII files.

In the following, you find detailed notes on formatting the files and the contents.

Tip

If you have iba measurement files at hand (e.g. created by *ibaPDA*), you can generate an ASCII template in an easy manner, by opening the iba measurement file with *ibaAnalyzer* and subsequently using the export function (Menu File - Export). If you can format your own ASCII files according to that scheme, the import should run without any problems.

2.2.2 General file structure

Row	Description
0	ID Time column (Time) and/or Signal-IDs ([M:K])
1	ID 'time' and/or signal names
2	ID 'sec' and/or signal units
3	\
4	> Section for Info fields/Technostrings
...	/
X	First data row (= first Samples), with time stamp where required
...	
Y	Measured values up to file end (each cell one sample per signal), with time stamp where required

Information like signal IDs, signal names, signal units and Info fields is optional, however, it is useful for a clear representation of the measured values in *ibaAnalyzer*.

In case the information is available in the file, but you do not want to use it, you have to deactivate the option in the import dialog.

For separating the individual information units within the rows, you should use only one common separator. A good choice is the TAB character (<TAB>, resp. <→I>) because this character is usually not part of the information. Also the semicolon (;) is a common delimiter. If you want to use comma or space as delimiter, you have to pay attention that these signs are not used in the data (e.g. space in signal names). The same is true for user defined delimiters.

The section between the first row and the beginning of the measurement value rows is not a fixed setting. However, we recommend the structure shown in table 1. Spaces, i.e. empty rows are allowed, but there must not be any empty rows within the data blocks with Info fields or measured values. As empty rows also must not contain delimiters, we recommend avoiding empty rows.

2.2.3 Notes on formatting

2.2.3.1 Signal IDs and time column identification (row 0)

The first row in the file should be reserved for the identification of a time row and the signal IDs.

Signal IDs

With the signal IDs, you can structure the measured values, according to the module-channel structure of *ibaAnalyzer*. Hence, you have access to a structured representation of the measured signals in the signal tree. If you do not use the signal ID, all signals in the signal tree are listed under the "0" module. Moreover, only by means of signal IDs, analog and digital signals can be distinguished.

The signal ID consists of the following components:

- [Module number**:**Channel number] for analog signals
- [Module number**.**Channel number] for digital signals
- [Module number:Channel number]_text for text signals

For an example illustrating the effect of the signal ID, please see the end of this chapter.

In the signal id, only the module's number is used for their identification. Module names can additionally be provided as info fields.

See chapter ↗ *Info fields*, page 14.

Identification time column

If you want to use a time stamp for the measured values in the file, the time information always has to be the first information in the measurement value rows. Please enter in this case "Time" in the first line and only thereafter - separated by delimiter - the signal IDs. This is necessary for maintaining the correct assignment of the signal IDs and measurement value rows.

Hence, you enter "time" as first information in the row with the signal name and "sec" as first information in the row with the units.

```
1  Time ⟶ [0:1] ⟶ [0:2] ⟶ [1:0] ⟶ [1.7]
2  time ⟶ Ana_Signal A ⟶ Ana_Signal B ⟶ Ana_Signal C ⟶ Dig_Signal D
3  sec ⟶ A ⟶ V ⟶ W
4
5  16.02.2015 17:51:01.430000 ⟶ 0 ⟶ 0 ⟶ 0 ⟶ 0
6  16.02.2015 17:51:02.430000 ⟶ 1 ⟶ 2 ⟶ 3 ⟶ 1
7  16.02.2015 17:51:03.430000 ⟶ 4 ⟶ 5 ⟶ 6 ⟶ 0
8  16.02.2015 17:51:04.430000 ⟶ 7 ⟶ 8 ⟶ 9 ⟶ 0
9
```

Fig. 3: ASCII file with time column (yellow) and signal IDs (green)

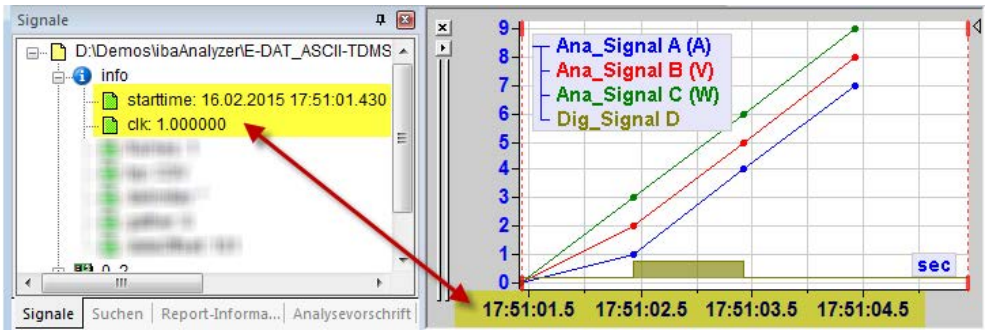
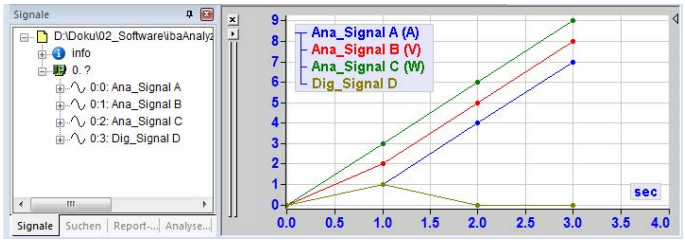


Fig. 4: Representation in ibaAnalyzer with starting time read out and computed timebase.

ASCII file without signal IDs

```
1  Ana_Signal A    Ana_Signal B    Ana_Signal C    Dig_Signal D
2  A    V    W
3  0    0    0    0
4  1    2    3    1
5  4    5    6    0
6  7    8    9    0
7
```

Result after import:

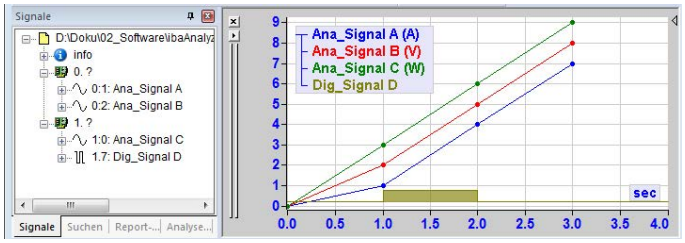


In the signal tree, signals are on one level. The digital signal "Dig_Signal D" is displayed like an analog signal.

ASCII file with signal IDs

```
1  [0:1]    [0:2]    [1:0]    [1.7]
2  Ana_Signal A    Ana_Signal B    Ana_Signal C    Dig_Signal D
3  A    V    W
4  0    0    0    0
5  1    2    3    1
6  4    5    6    0
7  7    8    9    0
```

Result after import:



Signals structured in the signal tree. Digital signal"Dig_Signal D" is displayed like a digital signal.

2.2.3.2 Signal names

If you want to transfer signal names to *ibaAnalyzer*, all signal names have to be in a row, preferably in the second row, but in any case in a row before the measurement value rows. In case there is no time column, the row begins with the first signal name. If there is a time column, the row begins with "time", followed by a delimiter and the signal names, also separated by a delimiter.

For the signal name you can use letters and numbers as well as hyphen, underscore and space. The selected delimiter should not be used in the signal name.

The signal names are displayed in *ibaAnalyzer* in the signal tree and the legend. They also serve as alternative reference in analysis functions (instead of the signal ID).

2.2.3.3 Signal units

If you want to transfer the physical signal units to *ibaAnalyzer*, all signal units have to be in one row, preferably in the third row, but in any case in a row previous to the rows for the measurement values.

In case there is no time column, the row begins with the first unit symbol. If there is a time column, the row begins with "sec", followed by a delimiter and the unit symbols, also separated by a delimiter.

The signal units are used in *ibaAnalyzer* for labeling the axes and in the legend. Digital signals do not have a unit, hence the respective position in the row remains empty. The number of delimiters has to equal the number of columns.

```
1  Time;[0:1];[0:2];[1:0];[1.7]
2  time;Ana_Signal A;Ana_Signal B;Ana_Signal C;Dig_Signal D
3  sec;A;V;W;
4
5  16.02.2015 17:51:01.430000;0;0;0;0
6  16.02.2015 17:51:02.430000;1;2;3;1
7  16.02.2015 17:51:03.430000;4;5;6;0
8  16.02.2015 17:51:04.430000;7;8;9;0
```

Fig. 5: Signal names (red) and signal units (blue)

2.2.3.4 Info fields

You can transfer additional information to *ibaAnalyzer* which is not part of the measurement samples. This information can consist of numbers as well as texts. You use the functions of the Infofield assignments and Technostrings in *ibaAnalyzer*. This additional information then is displayed in the "Info" branch in the signal tree and can be used in the analyses with appropriate functions, e.g. as text channel.

You can enter any number of Info fields. In one row, only one Info field may be defined. The Info fields should be placed coherently after the rows we have described above and prior to the beginning of the measurement value rows.

As first information (first column) the name of the Info field is found in the row. After a delimiter, in the second column, the content of the Info field can be found (numbers, text or alphanumeric string).

```
Time [0:1] [0:2] [1:0] [1.7]
time Ana_Signal A Ana_Signal B Ana_Signal C Dig_Signal D
sec A V W

Infofield_A 10
Infofield_B 0.5
Technostring_1 Text file with ibaAnalyzer time format

16.02.2015 17:51:01.430000 0.0 0.0 0.0 0
16.02.2015 17:51:02.430000 1.25 2.5 3.75 1
16.02.2015 17:51:03.430000 4.25 5.50 6.75 0
16.02.2015 17:51:04.430000 7.25 8.5 9.75 0
```

Fig. 6: Example for 3 Info fields

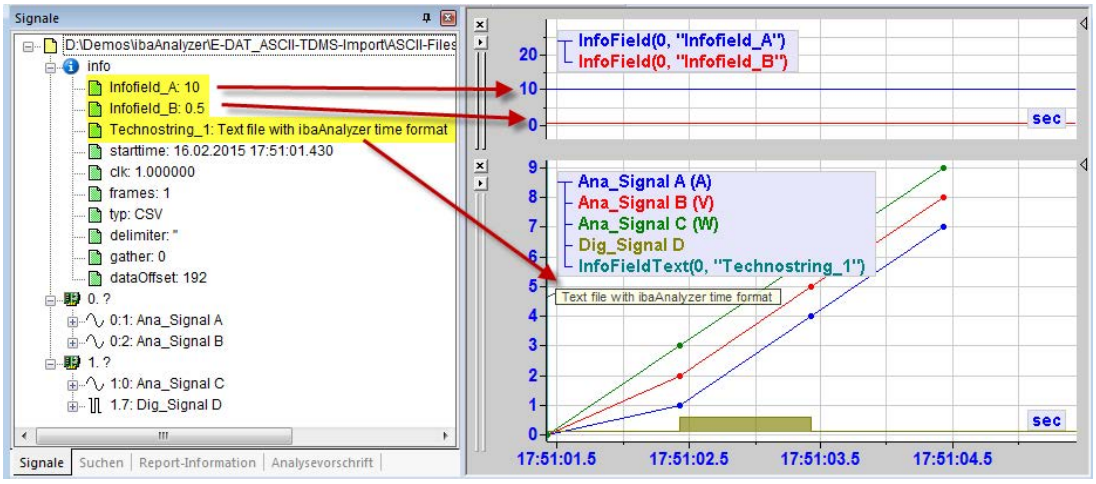


Fig. 7: Display of the Info fields in ibaAnalyzer

Another important purpose of the info field section is the possibility to name the modules.

For this function the syntax of the info fields is defined:

`Module_name_` *Index*;My modul name

Index is the number of the module as it has been defined in the signal id (e.g. 0, 1, ...). After a delimiter (e.g. TAB or semicolon) follows the desired clear text name of the module.

The following picture shows as example a text file with 2 modules (No. 0 and 1), represented in *ibaAnalyzer*. The inset shows the text file with the module name definition. The module names are displayed in the signal tree accordingly.

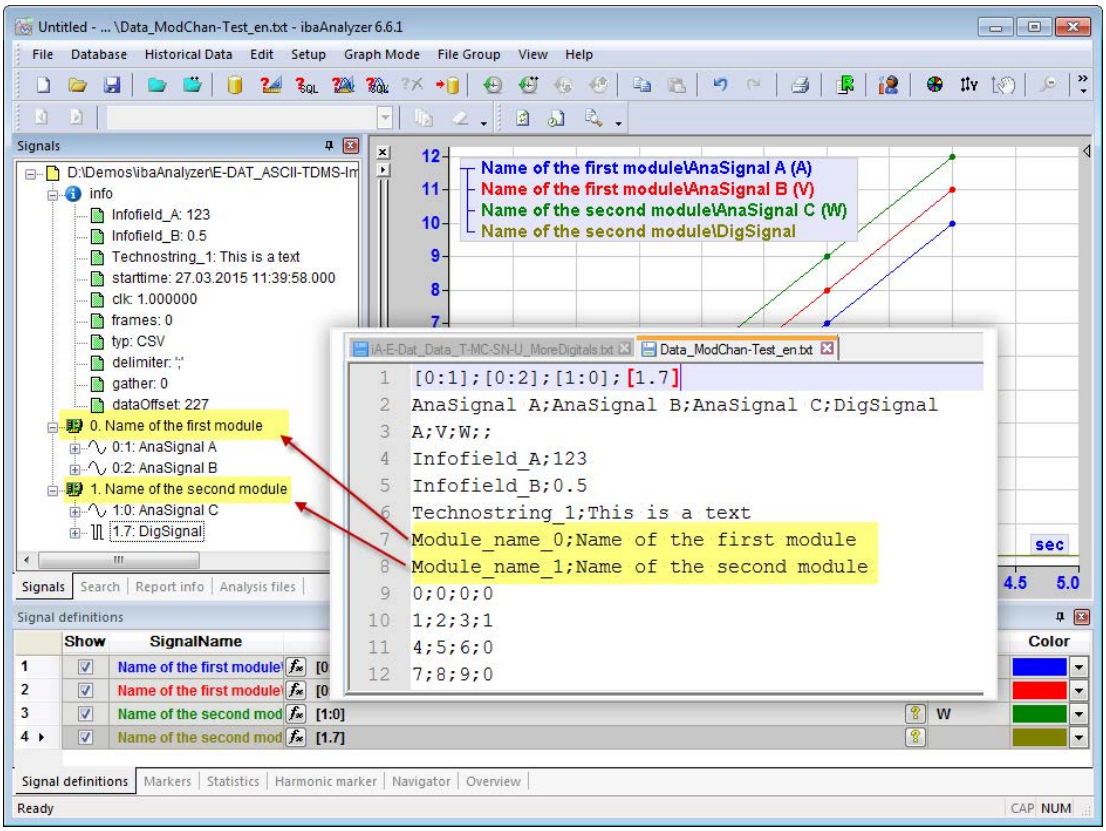


Fig. 8: Definition of module names in the text file and the result in the signal tree of ibaAnalyzer

2.2.3.5 Data rows

The data rows contain the time stamps and measurement values.

If there is no time column, the row begins with the measurement value of the first signal. If there is a time column, the row begins with a time stamp (date, time) followed by a delimiter and the measurement values of all signals at this point of time, also separated by a delimiter.

Note

As decimal sign for floating point values you have to use the sign which is set in the Windows Control Panel *Regional Settings - Formats - Additional Settings - Decimal Symbol*

2.2.3.6 Time stamp

In case there is a time stamp for the measurement values, it has to be placed at the beginning of each data row.

When formatting the time information, you should pay attention that it either corresponds to the *ibaAnalyzer* time format (dd.mm.yyyy hh:mm:ss.s) or the settings in the Windows control panel.

In case you select the time column option "Non equidistant samples/ ibaAnalyzer time format", the time information has to be formatted according to the settings in the Windows control pan-

el. If this is not the case, *ibaAnalyzer* cannot interpret the time information and does not display the measurement values.

Note

The extension for the microseconds made up of 6 digits can be added to the time information following a decimal separator. For the ibaAnalyzer time format this separator has to be a dot, for the local time format it can be a dot or a comma.

Example for setting "German (Germany)" and the use of the ibaAnalyzer time format.

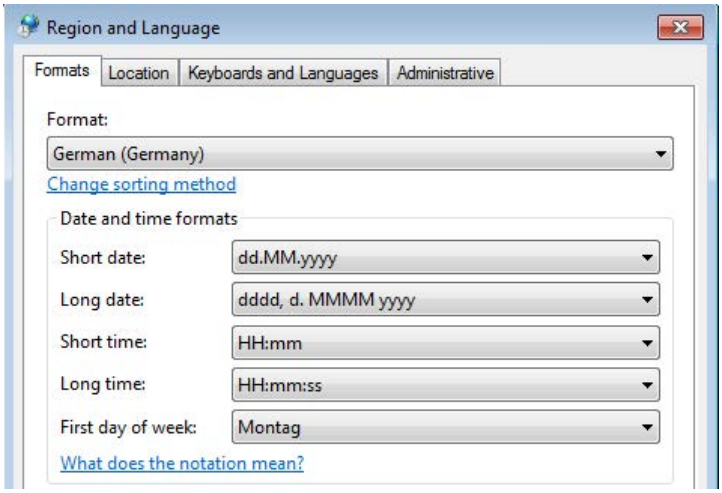


Fig. 9: Windows control panel, Region and Language, Format Germany

```
1  Time [0:1] [0:2] [1:0] [1.7]
2  time Ana_Signal A Ana_Signal B Ana_Signal C Dig_Signal D
3  sec A V W
4
5  Infofield_A 10
6  Infofield_B 0.5
7  Technostring_1 Text file with ibaAnalyzer time format
8
9  16.02.2015 17:51:01.430000 0,0 0,0 0,0 0
10 16.02.2015 17:51:02.430000 1,25 2,5 3,75 1
11 16.02.2015 17:51:03.430000 4,25 5,50 6,75 0
12 16.02.2015 17:51:04.430000 7,25 8,5 9,75 0
```

Fig. 10: Template ASCII File, German

Example for setting "English (United States)" and use of the local (US-) time format

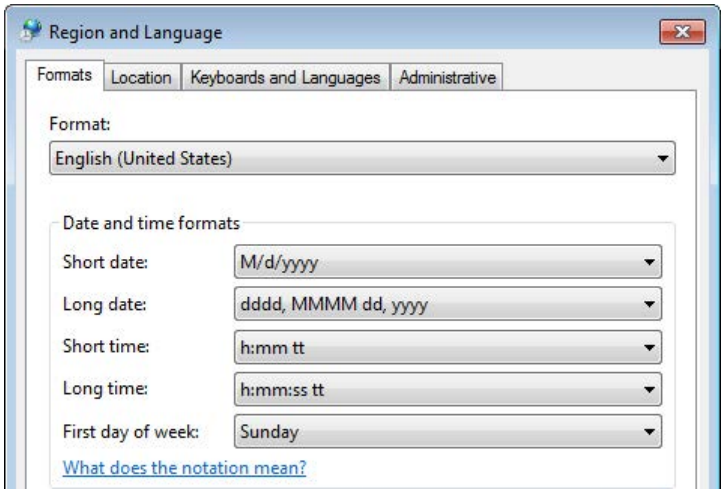


Fig. 11: Windows Control Panel, Region and Language, Format English (United States)

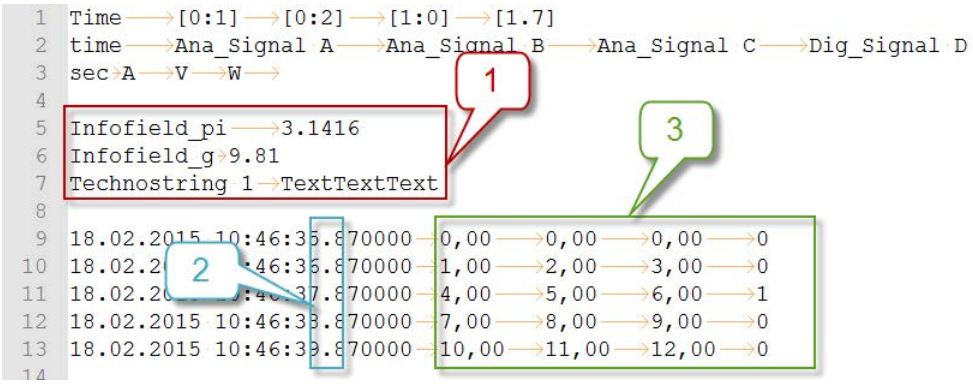
```
1  Time→[0:1]→[0:2]→[1:0]→[1.7]
2  time→Ana_Signal A→Ana_Signal B→Ana_Signal C→Dig_Signal D
3  sec→A→V→W→
4
5  Infofeld_A→10
6  Infofeld_B→0.5
7  Technostring_1→This is a file with US date-time format
8
9  2/16/2015 5:51:01.430000 pm→0.0→0.0→0.0→0
10 2/16/2015 5:51:02.430000 pm→1.5→2.5→3.75→1
11 2/16/2015 5:51:05.430000 pm→4.5→5.50→6.75→0
12 2/16/2015 5:51:07.430000 pm→7.5→8.5→9.75→0
```

Fig. 12: Template ASCII File, English

2.2.3.7 Notes on the use of decimal separators

Using the correct decimal separators is essential for importing numerical data correctly. The data is interpreted depending on the type, the system settings and partially on the kind of import.

For the import, we differentiate between three different types of data or sections:



1 Info field assignments and Technostrings
(Technostrings only available in data files created by *ibaPDA* <v7.0)

2 Time information with microseconds

3 Measurement values (analog values)

Depending on the section, you have to pay attention to the following points when using decimal separators.

Info field assignments and Technostrings

Information in Info fields and Technostrings are generally interpreted as text (string) by *ibaAnalyzer*. For displaying the content of these values in *ibaAnalyzer*, special functions are used:

- InfoField: Generates a numeric signal
- InfoFieldText: Generates a text channel

If you want to assign a numerical value (floating point) to an Info field, you always have to use a dot as decimal separator **independently of the Windows regional settings**.

If you use a comma instead, the content of the info field is always interpreted and displayed as text (text channel).

You can easily understand it by dragging in *ibaAnalyzer* an Info field from the signal tree to the trend graph or by double-clicking it. If there is a number with a decimal point in the Info field, the *InfoField* function will be applied automatically. If it is a number with a decimal comma, the *InfoFieldText* function will be used automatically.

In case you use the *InfoField* function on purpose, only the digit before the comma is interpreted as value.

Time information with microseconds

If you want to import a highly precise time stamp with the measurement values, you can add after the common time information hh:mm:ss up to 6 positions (microseconds) after a decimal separator.

It only depends on the desired time format and/or the time column option, which decimal separator you have to use. The regional Windows settings are irrelevant.

Time column option (Import dialog)	Decimal separator
None	None
ibaAnalyzer export (equidistant values)	Dot
Non equidistant values, ibaAnalyzer date and time format	Dot
Non equidistant samples, local time format:	Dot or comma

Measurement values

For analog values in the floating point format, you always have to use the decimal separator which is preselected in the Windows regional settings.

2.3 TDMS files

In order to be able to read files in TDMS-format, the files must have the suffixes *.tdm* or *.tdms*.

Important note

If you want to use the TDMS functionality, you need to install an additional program on the computer on which you want to open the TDMS files with *ibaAnalyzer*: *ibaTDMSBundleInstall.exe*

You find this file packed as ibaTDMSBundleInstall.zip on the data medium “iba Software & Manuals” in the directory "...01_Software\ibaAnalyzer\TDMS-Extraction

The channel groups contained in this file type are displayed in *ibaAnalyzer* as modules. The related channels are arranged below the modules. File properties are transferred in *ibaAnalyzer* to the Info fields and arranged in the Info branch of the signal tree. Channel properties are transferred to Channel Info fields and displayed under the respective channel in the signal tree. Channel group properties are transferred to module info fields and can be interpreted using the functions *ModuleInfoField* or *ModuleInfoFieldText* respectively. The time stamps of the measurement values are determined using the following variables:

- Start time property of the file (*datetime*)
- Channel properties *wf_increment*, *wf_start_offset* and *wf_start_time*

Other channel properties are used as follows:

- *name*: Channel or signal name
- *unit_string*: If available, then as unit for the measured values
- *description*: If available, then as comment 1 of a channel
- *wf_xname*: If available, the value of this channel is displayed in another physical domain than that of time. In case this parameter is not available or contains a forbidden value, the measurement values of the channel are displayed in a time-based manner with seconds being the basic unit.

  The following values are permitted.

- *length*: Length-based signals, the standard length unit selected in *ibaAnalyzer* is the basic unit
- *frequency*: Frequency based signals, the basic unit is Hertz
- *invlength*: Inversely length-based signals, basic unit is 1/standard length unit

As the TDMS measurement file format does not supply a clear indicator which shows if a signal is to be interpreted as digital signal or not, *ibaAnalyzer* uses other ways for classifying a signal as digital signal. A signal is a digital signal, if it meets one of the following conditions:

- A channel property "*iba_digital*" of the 8-Bit Integer type is available and has a value unequal to 0.
- A channel property "*unit_string*" is available and has the value "*bit*".
- The name of the channel ends with "*[bit]*"

2.4 Apache Parquet

Apache Parquet is a column-oriented, binary data format which provides efficient data compression and different encodings. Due to its columnar structure and the possibility to add meta data the file format, it can resemble the iba .dat file format. Also due to the comparable storage size we recommend these files as an interchange format to external systems.

Compression

The Apache Parquet format offers different compression methods. *ibaAnalyzer* supports plain encoding (Uncompressed), Snappy, Gzip, Brotli, LZ4, and ZStandard.

2.4.1 Files created by ibaAnalyzer

With the export dialog of *ibaAnalyzer* or the data extractor it is possible to create parquet files from *ibaAnalyzer*. These files can of course be opened again by *ibaAnalyzer*. The data in the output file are structured as follows:

- A channel (or expression) corresponds to a Parquet column
- The module structure available in iba .dat files has no direct pendant in Parquet and is therefore mapped using meta data.
- All info fields are stored as Parquet meta data.

The correct structure of data enables *ibaAnalyzer* to restore the complete file structure when opening the extracted Parquet file. In this case the modules are present and all (non-empty) info fields are available.

Note that the info fields get sorted alphabetically after the re-import. The necessary meta data to resemble such a structure in your custom parquet files is described below.

Other documentation

For further information about the data extractor in *ibaAnalyzer*, please refer to the *ibaAnalyzer-File-Extract* manual.

2.4.2 Open generic parquet files

Using *ibaAnalyzer-E-Dat* it is possible to open generic parquet files.

In case, if there is no metadata with key “clk”, the input of a time base is required. This time base is then used for the imported data from the individual columns.

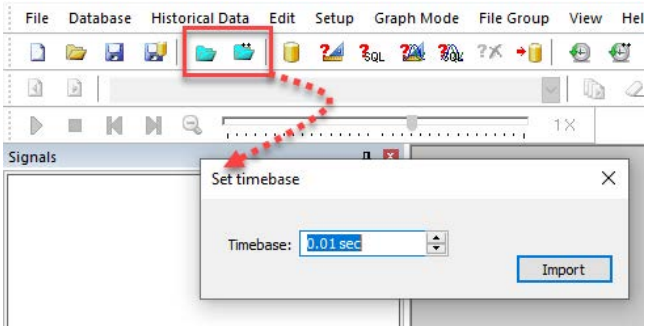


Fig. 13: Prompt for entering a time base when opening a file without time column

On import of generic files all signals will be put into a single module “0” with a straight-through enumeration. The column names are used as signal name.

Datatypes

The Parquet format defines many data types; not all of them are supported by *ibaAnalyzer*. Below is a list of supported types with the corresponding resulting type in *ibaAnalyzer*. The supported data types can be extended on request.

Parquet data type	Internal storage in ibaAnalyzer	Comment
Type::**STRING**	**string**	
Type::**BOOL**	**bool**	
Type::**FLOAT**	**float**	
Type::**HALF_FLOAT**	**float**	no loss of precision; some waste of memory in favor of simplicity
Type::**DOUBLE**	**double**	
Type::**INT8** Type::**UINT8** Type::**INT16** Type::**UINT16**	**float**	no loss of precision; some waste of memory in favor of simplicity
Type::**INT32** Type::**UINT32**	**double**	
Type::**TIMESTAMP** (based on int64)	**double**	Possible loss of precision in favor of simplicity Please note that there is no proper way in *ibaAnalyzer* to display time values at Y axis. They appear just as very big numbers without an obvious meaning .

Table 1: Parquet data types and corresponding ibaAnalyzer data types

2.4.3 Metadata interpreted by ibaAnalyzer

When opening a parquet file with *ibaAnalyzer*, several meta data are interpreted in a special way. This mechanism is used to restore the original file structure for exported dat-files. By using the correct meta data you can structure your parquet files when opened in *ibaAnalyzer*.

Time column

The column “time” (e.g. for files exported from *ibaAnalyzer* with the corresponding option) is ignored, if the meta data “clk” and “starttime” are present. These values are used to determine the sample rate and the start of the file.

If this meta data is not present the time column will be treated like a normal signal and will therefore also be available in *ibaAnalyzer* as a “signal”.

Column names

Usually, *ibaAnalyzer* interprets the column name as signal name. The display name can be overwritten by using the correct signal-level meta data (see below).

2.4.3.1 File-level meta data

All plain key-value pairs are interpreted as file-level info fields. Some reserved keywords are interpreted by *ibaAnalyzer*:

- “clk” defines the sample rate of the signals in seconds
- “starttime” sets the start time of the file. The values need to be formatted in epoch time.
- “module_name_n” sets the display name of module number “n”.

2.4.3.2 Module-level meta data

All keywords beginning with “M[n]” are interpreted as module-level info fields for the module number n. These data are not interpreted in a special way, they are just sorted under the corresponding module in the signal tree.

2.4.3.3 Signal-level meta data

Keywords beginning with “[column_name]”, where column_name is the name of the corresponding column in the parquet file, are interpreted as signal-level info fields. The following fields are interpreted:

- “[column_name]name”
  sets the displayed signal name in *ibaAnalyzer*
- “[column_name]unit”
  sets the unit displayed in the signal grid
- “[column_name]$PDA_comment1”
  sets the comment 1 displayed in the signal grid. Similar for comment 2 ($PDA_comment2)
- “[column_name]Lengthbased”
  the presence of this field indicates that the column contains length-based data
- “[column_name]LengthBase”
  in case of length-based data the sample rate in meter is specified here
- “[column_name]Nr”
  sets the module and signal number. Here the *ibaAnalyzer* notation is used.

Examples:

- “3.5” creates a digital signal in module 3 with signal number 5
- “4:2” indicates an analog signal in module 4 with signal number 2
- “2:1:1” indicates the first subchannel of the signal number 1 in module 2

2.5 Matlab

The software MATLAB® of company MathWorks® provides its own (binary) data format with file extension ".mat". With *ibaAnalyzer* it is possible to open .mat files if they have a specific structure which is described below.

Tip

Data with the correct structure can also be generated with the data extractor in *ibaAnalyzer*. For more information about the data extractor in *ibaAnalyzer*, please refer to the *ibaAnalyzer-File-Extract* manual.

2.5.1 Data structure

The software Matlab supports different datatypes and structures. *ibaAnalyzer* supports .mat files which contain so-called "struct" objects. A "struct" object in Matlab can contain any type of data, especially also other "struct" objects.

ibaAnalyzer supports nested "struct" objects in the following form:

Struct "filename"

- Struct "fileinfo"
- Struct "Module_name_1"
  - Struct "moduleinfo"
  - Struct "signalname_1"
  - Struct "signalname_2"
  - ...
  - Struct "signalname_n"
- Struct "module_name_2"
  - Struct "moduleinfo"
  - Struct "signalname_1"
  - …
- ...
- Struct "module_name_n"

Using such nested structures, you can create files sorted by modules and signals similar to the iba DAT-files.

The different struct objects can contain various information which will be displayed in *ibaAnalyzer* either as signals or as info fields on file-level, module-level, or signal-level, respectively.

2.5.1.1 Struct "filename"

This is the root structure. It contains only other structures containing info fields or modules. Any other data present in this structure will be ignored.

The name of the structure will be used as filename displayed in the *ibaAnalyzer* signal tree.

2.5.1.2 Struct "fileinfo"

This structure needs the fixed name “fileinfo” to be interpreted correctly.

Any numerical or textual key-value-pairs contained will be displayed as info-fields in *ibaAnalyzer*.

A special field “clk” indicates the sampling time (in s) of the signal data.

Further, a field “starttime” is needed to fix the time stamp of the first data point.

2.5.1.3 Struct "module_name"

The name of the structure will be used as module name when displayed in *ibaAnalyzer*. If the structure “moduleinfo” contains a field “name” this value will be used instead for the name.

Like for the root structure, these structures contain only other structures with module-level info fields or signals. Other information will be ignored.

2.5.1.4 Struct "moduleinfo"

This structure needs the fixed name “moduleinfo” to be interpreted correctly. It can be present in any module.

Like on the file-level, any numerical or textual key-value-pairs contained will be displayed as corresponding module-level info fields in *ibaAnalyzer*.

2.5.1.5 Struct "signal_name"

The name of the structure will be used as signal name when displayed in *ibaAnalyzer*. If a field “name” is present within the structure, this will be used instead.

Any other key-value pair will be shown as signal-level info fields in *ibaAnalyzer*.

The structures on this level also contain the actual signal data. Every signal structure should contain an array with the fixed name “data” which contains measurement data. These values will be displayed as signal in *ibaAnalyzer*.

2.5.2 Metadata interpreted by ibaAnalyzer

Like indicated above, several meta data on the different levels are interpreted in a special way by *ibaAnalyzer*. This mechanism is used to restore the original file structure for exported DAT-files.

By using the correct meta data you can structure your Matlab files when opened in *ibaAnalyzer*.

2.5.2.1 File-level meta data

All plain key-value pairs in the structure “fileinfo” are interpreted as file-level info fields. Some reserved keywords are interpreted by *ibaAnalyzer* as follows:

- “clk” defines the sampling time of the signals in seconds
- “starttime” sets the start time of the file. The values needs to be a string formatted as “dd.mm.yyyy hh:mm:ss.msmsms”.

The field “Time” (e.g. for files exported from *ibaAnalyzer* with the corresponding option) contains a column vector with the time stamps for the signals. The field is ignored, if the meta data “clk” and “starttime” are present. These values are used to determine the sample rate and the start of the file.

2.5.2.2 Module-level meta data

All key-value pairs present in a “moduleinfo” structure are interpreted as module-level info fields for the corresponding module number. Most of the fields are not interpreted in a special way, they are just sorted under the corresponding module in the signal tree.

The only exception is the field “ModuleID” which is used to store the module number displayed in *ibaAnalyzer*.

2.5.2.3 Signal-level meta data

Key-value pairs present in the “signal” structures are interpreted as signal-level info fields. The only exception is the field “data” which contains the measurement values.

The following fields are interpreted in a special way:

- "SignalID"
  sets the signal number within the module
- “name” sets the displayed signal name in *ibaAnalyzer*
- “unit” sets the unit displayed in the signal grid
- “PDA_comment1” sets the comment 1 displayed in the signal grid. Similar for comment 2 (PDA_comment2)
- "PDA_TBase"
  sets a sampling time different from the global “clk” value
- “Lengthbased” the presence of this field indicates that the column contains length-based data
- “LengthBase” in case of length-based data the sample rate in meter is specified here

3 Appendix: Examples for working with spread sheet programs

The following examples show different possibilities of presenting data as tables, beginning with the simple measurement value table up to a comprehensive table with time stamp and Info fields.

In every example you find the following components:

- Table structure
- Table with example values
- Exported text file with tabulator as delimiter and the matching *ibaAnalyzer* import settings
- Exported CSV file with semicolon as delimiter and the matching *ibaAnalyzer* import settings

In the examples, we have always used the signal IDs.

Tip

Please avoid empty rows in the tables as for the export in text or CSV files in the empty rows delimiters are written for every column.

You have to delete these delimiters from the files using a text editor before reading them into *ibaAnalyzer*. The same is true for empty columns in the Info fields section.

```
1  Time;[0:1];[0:2];[1:0];[1.7]
2  time;Ana_Signal A;Ana_Signal B;Ana_Signal C;Dig_
3  sec;A;V;W;
4  ;;;;
5  Infofield_pi;3,1416;;;
6  Infofield_g;9,81;;;
7  Technostring 1;TextTextText;;;
8  ;;;;
9  18.02.2015 10:46:35.870000;0,00;0,00;0,00;0,00
10 18.02.2015 10:46:36.870000;1,00;2,00;3,00;0,00
```

File export with redundant delimiters

Empty rows containing delimiters lead to an extended x-axis in the graph view although there are no values.

Empty rows with delimiters in the Info field section cannot be interpreted by *ibaAnalyzer*.

3.1 Measurement value table with signal names and units without time

Table structure

	Column 1	Column 2	...	Column m
Row 0	[M:C]	[M:C]	[M:C]	[M:C]
Row 1	Signal name 1	Signal name 2	Value	Value
Row 2	Unit	Unit	Value	Value
Row 3	Value	Value	Value	Value
Row 4	Value	Value	Value	Value
Row 5	Value	Value	Value	Value
Row 6	Value	Value	Value	Value
Row 7	Value	Value	Value	Value
Row 8	Value	Value	Value	Value
Row 9	Value	Value	Value	Value
Row 10	Value	Value	Value	Value
Row 11	Value	Value	Value	Value
...	...	...	...	...
Row n	Value	Value	Value	Value

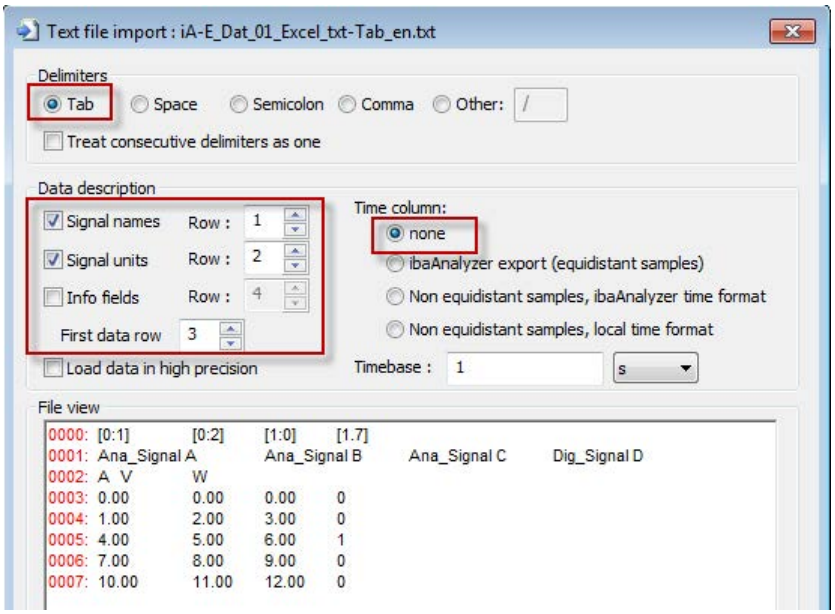
Table with values

[0:1]	[0:2]	[1:0]	[1.7]
Ana_Signal A	Ana_Signal B	Ana_Signal C	Dig_Signal D
A	V	W	
0,00	0,00	0,00	0
1,00	2,00	3,00	0
4,00	5,00	6,00	1
7,00	8,00	9,00	0
10,00	11,00	12,00	0

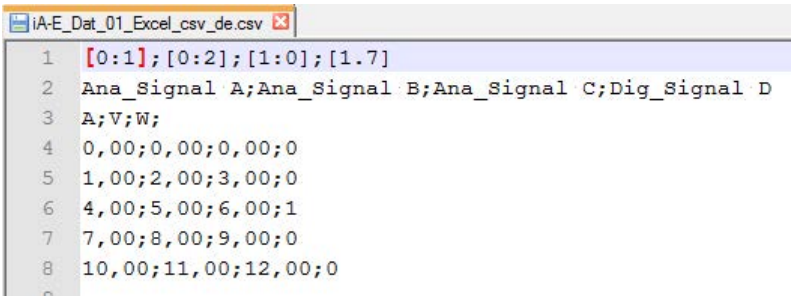
Excel export as text file with tab and ibaAnalyzer import settings

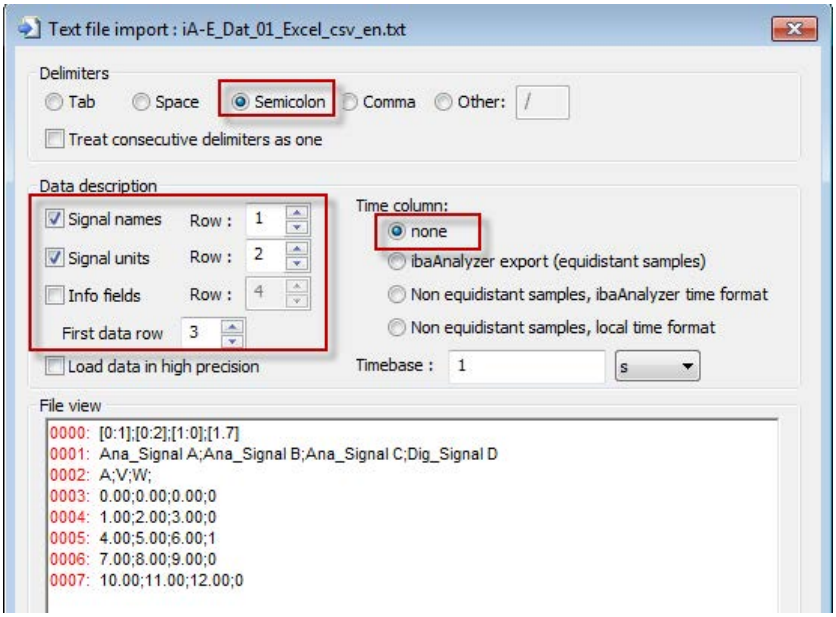
iA-E_Dat_01_Excel_txt-Tab_de.txt

```
[0:1]→[0:2]→[1:0]→[1.7]
Ana_Signal A→Ana_Signal B→Ana_Signal C→Dig_Signal D
A→V→W→
0,00→0,00→0,00→0
1,00→2,00→3,00→0
4,00→5,00→6,00→1
7,00→8,00→9,00→0
10,00→11,00→12,00→0
```



Excel export as CSV file with semicolon and ibaAnalyzer import settings





3.2 Measurement value table with time, signal names and units

Table structure

	Column 1	Column 2	...	Column m
Row 0	Time	[M:C]	[M:C]	[M:C]
Row 1	time	Signal name 1	Signal name 2	Signal name n
Row 2	sec	Unit	Unit	Unit
Row 3				
Row 4	Timestamp 0	Value	Value	Value
Row 5	Timestamp 1	Value	Value	Value
Row 6	Timestamp 2	Value	Value	Value
Row 7	Timestamp 3	Value	Value	Value
Row 8	Timestamp 4	Value	Value	Value
Row 9	Timestamp 5	Value	Value	Value
Row 10	Timestamp 6	Value	Value	Value
Row 11	Timestamp 7	Value	Value	Value
...	...	...	...	...
Row n	Timestamp n	Value	Value	Value

Table with values

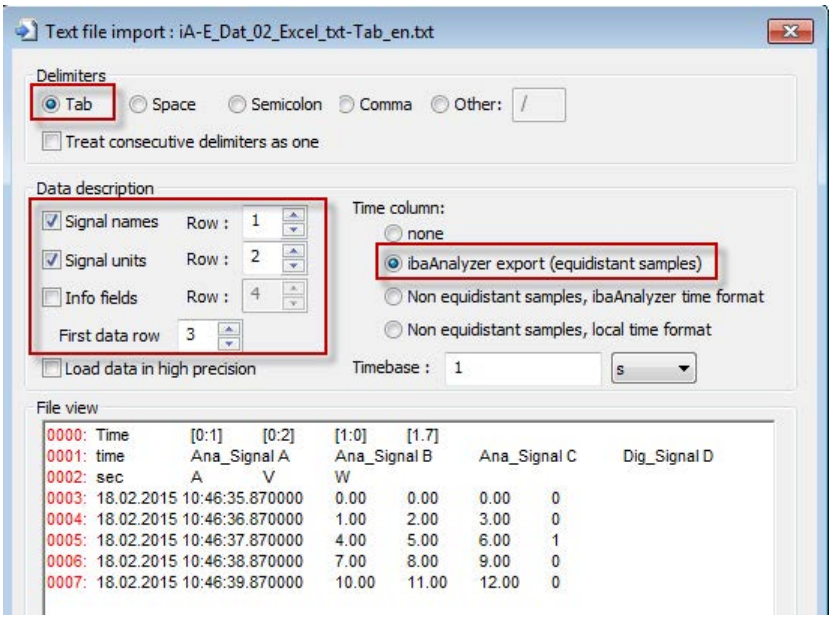
Time	[0:1]	[0:2]	[1:0]	[1.7]
time	Ana_Signal A	Ana_Signal B	Ana_Signal C	Dig_Signal D
sec	A	V	W	
18.02.2015 10:46:35.870000	0,00	0,00	0,00	0
18.02.2015 10:46:36.870000	1,00	2,00	3,00	0
18.02.2015 10:46:37.870000	4,00	5,00	6,00	1
18.02.2015 10:46:38.870000	7,00	8,00	9,00	0
18.02.2015 10:46:39.870000	10,00	11,00	12,00	0

Avoid empty rows!

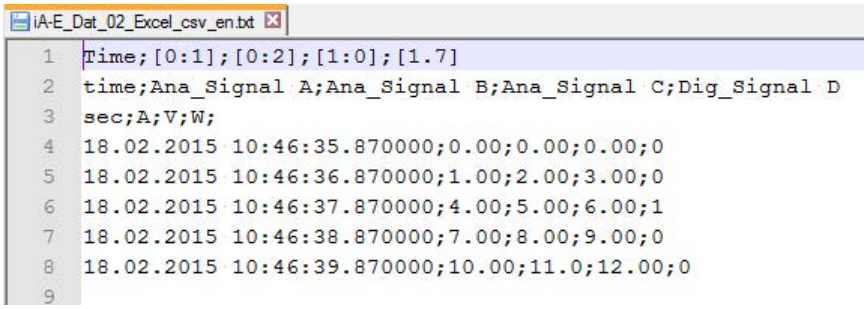
Excel export as text file with tab and ibaAnalyzer import settings

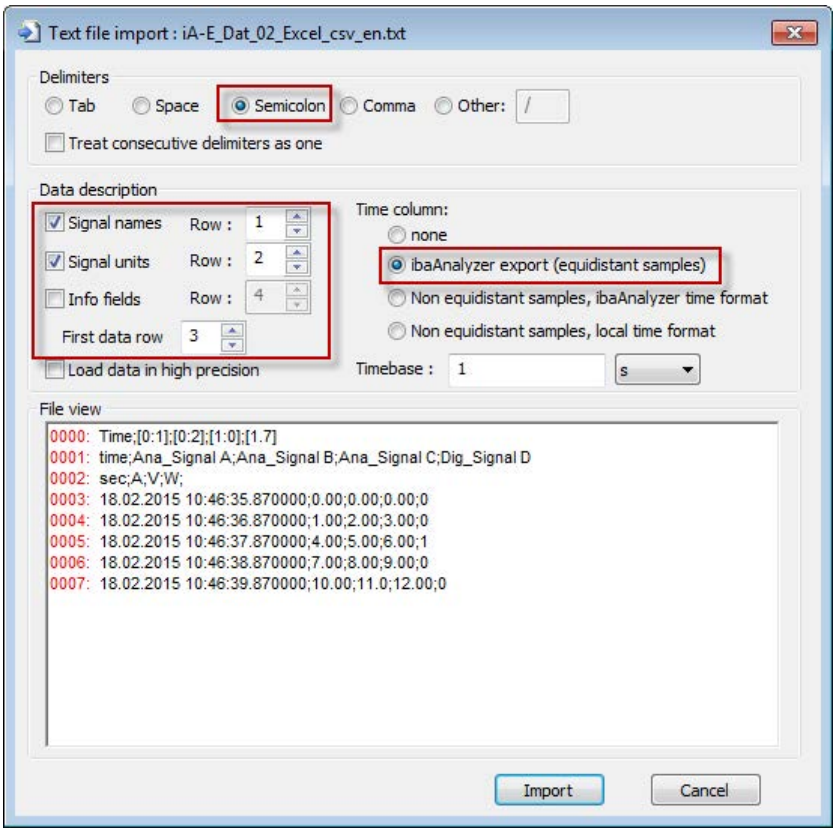
iA-E_Dat_02_Excel_txt-Tab_en.txt

```
1  Time→[0:1]→[0:2]→[1:0]→[1.7]
2  time→Ana_Signal A→Ana_Signal B→Ana_Signal C→Dig_Signal D
3  sec→A→V→W→
4  18.02.2015 10:46:35.870000→0.00→0.00→0.00→0
5  18.02.2015 10:46:36.870000→1.00→2.00→3.00→0
6  18.02.2015 10:46:37.870000→4.00→5.00→6.00→1
7  18.02.2015 10:46:38.870000→7.00→8.00→9.00→0
8  18.02.2015 10:46:39.870000→10.00→11.00→12.00→0
```



Excel export as CSV file with semicolon and ibaAnalyzer import settings





3.3 Measurement value table with time, signal names, units and Info fields

Table structure

	Column 1	Column 2	...	Column m
Row 0	Time	[M:C]	[M:C]	[M:C]
Row 1	time	Signal name 1	Signal name 2	Signal name n
Row 2	sec	Unit	Unit	Unit
Row 3				
Row 4	Infofield 1	Content Info/String 1		
Row 5	Infofield 2	Content Info/String 2		
Row 6	...	...		
Row 7	Infofield X	Content Info/String X		
Row 8				
Row 9	Timestamp 0	Value	Value	Value
Row 10	Timestamp 1	Value	Value	Value
Row 11	Timestamp 2	Value	Value	Value
...	...	...	...	...
Row n	Timestamp n	Value	Value	Value

Table with values

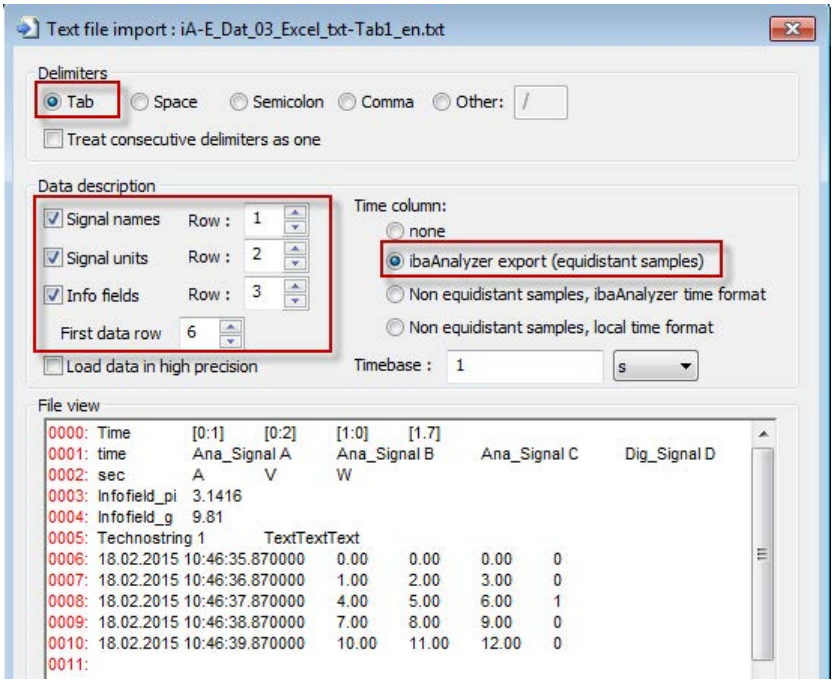
Time	[0:1]	[0:2]	[1:0]	[1.7]
time	Ana_Signal A	Ana_Signal B	Ana_Signal C	Dig_Signal D
sec	A	V	W	
Infofield_pi	3.1416			
Infofield_g	9.81			
Technostring 1	TextTextText			
18.02.2015 10:46:35.870000	0.00	0.00	0.00	0
18.02.2015 10:46:36.870000	1.00	2.00	3.00	0
18.02.2015 10:46:37.870000	4.00	5.00	6.00	1
18.02.2015 10:46:38.870000	7.00	8.00	9.00	0
18.02.2015 10:46:39.870000	10.00	11.00	12.00	0

Avoid empty rows!

Excel export as text file with tab and ibaAnalyzer import settings

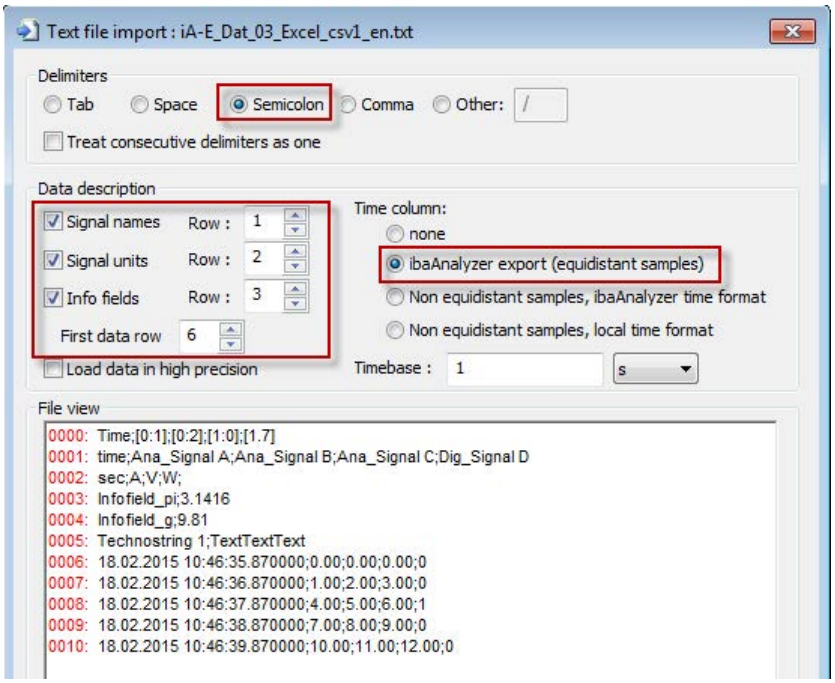
iA-E_Dat_03_Excel_txt-Tab1_en.txt

```
Time	[0:1]	[0:2]	[1:0]	[1.7]
time	Ana_Signal A	Ana_Signal B	Ana_Signal C	Dig_Signal D
sec	A	V	W
Infofield_pi	3.1416
Infofield_g	9.81
Technostring 1	TextTextText
18.02.2015 10:46:35.870000	0.00	0.00	0.00	0
18.02.2015 10:46:36.870000	1.00	2.00	3.00	0
18.02.2015 10:46:37.870000	4.00	5.00	6.00	1
18.02.2015 10:46:38.870000	7.00	8.00	9.00	0
18.02.2015 10:46:39.870000	10.00	11.00	12.00	0
```



Excel export as CSV file with semicolon and ibaAnalyzer import settings

```
1  Time;[0:1];[0:2];[1:0];[1.7]
2  time;Ana_Signal A;Ana_Signal B;Ana_Signal C;Dig_Signal D
3  sec;A;V;W;
4  Infofield_pi;3.1416
5  Infofield_g;9.81
6  Technostring 1;TextTextText
7  18.02.2015 10:46:35.870000;0.00;0.00;0.00;0
8  18.02.2015 10:46:36.870000;1.00;2.00;3.00;0
9  18.02.2015 10:46:37.870000;4.00;5.00;6.00;1
10 18.02.2015 10:46:38.870000;7.00;8.00;9.00;0
11 18.02.2015 10:46:39.870000;10.00;11.00;12.00;0
```



4 Support and contact

Support

Phone: +49 911 97282-14

Email: support@iba-ag.com

Note

If you need support for software products, please state the number of the license container. For hardware products, please have the serial number of the device ready.

Contact

Headquarters

iba AG
Koenigswarterstrasse 44
90762 Fuerth
Germany

Phone: +49 911 97282-0

Email: iba@iba-ag.com

Mailing address

iba AG
Postbox 1828
D-90708 Fuerth, Germany

Delivery address

iba AG
Gebhardtstrasse 10
90762 Fuerth, Germany

Regional and Worldwide

For contact data of your regional iba office or representative please refer to our web site:

www.iba-ag.com