



File Format Specification
Document Version 1.0

Questions?

Contact [testingsolutions](#):



www.testingsolutions.de

internet



info@testingsolutions.de

e-mail



+49 (0) 8191 305 202

Phone



+49 (0) 8191 305 244

FAX

[testingsolutions GmbH&Co. KG](#)



Schwaighofstr. 4a
D-86899 Landsberg/Lech
Germany

Mail



bdf
Version 5.0.6
File Format Specification
Document Version 1.0
printed Oktober 2004

Document Info



This document is protected by
international law.

[testingsolutions](#), 2004

Linux is a registered trademark of Linus Torvalds

Windows is a registered trademark of the Microsoft Corporation

MATLAB is a registered trademark of The Mathworks.

Other names mentioned in this manual may be registered trademarks of their owners and must be treated as trademarks.

Modification and distribution of this manual in electronic, printed or any other form without the prior written authorization is of the copyright owner is prohibited.

This documentation is provided as is without any warranty for completeness and integrity. The author does not assure freedom from errors. This manual or parts of this manual may be modified at any time without prior notice.

Contents

Release Notes.....	1
bdf File Structure.....	3
File Header.....	4
Channel Information.....	5
Data section.....	5
Timetable.....	5
Supported Data Types.....	7
Unsigned Integer Data Types.....	8
Signed Integer Data Types.....	8
Floating Point Data Types.....	8
Discrete Value Data Types.....	9
Data Compression.....	11
Compression Methods.....	12
Physical File Structure.....	13
File Header.....	14
Header Variables.....	20
Channel Information.....	21
Data Section.....	24
Timetable Section.....	26

Release Notes



The Information in this manual refers to bdf-file release 5.0.6 only!

bdf File Structure

The information in the file is stored in binary form. A typical case of application is the storage of multiple channels of data over a common time base, as they are generated e.g. by a data acquisition system. The File consists of four major parts that are sequentially stored:

File Header	Channel Information	Data Section	Time-table
-------------	---------------------	--------------	------------

File Header

The file header contains the information on the physical structure of the bdf-file and basic information about the stored data. The first five bytes indicate the release number of the used file format which determines the interpretation of the following header structure. Further information identifies the hard- and software configuration of the creating operating system, as the internal data formats may vary.

The time of file creation, start and end time of the data are also stored in the file header. All times used in a bdf-file header must be given in serial time format indicating the elapsed days since January 1st, 0000 in UTC. For accuracy reasons, all mathematical operations involving serial times should always use double precision values. The bdf-Toolboxes offers various functions to convert e.g. date strings into serial time values. Additional information about the local time zone where the data was gathered may be added.

Also specified in the header section is the number of channels that contain the data. The file header may also include user data that is attached to the header structure.

Channel Information

The channel information describes the channels that are stored inside the data section. It contains essential information for every used channel, such as its name, type of the data, number of occurrences in one data block (see below) and its position in this block. Additional information, that is not mandatory for reading the file, but may be required by applications using the bdf-file (e.g. `testeval`), can be stored in user variables in an identical manner as in the header section.

Data section

The data section contains the values of the channels. The values can be grouped into data blocks. All data blocks must have the same preset interval length, but every block can have an individual starting time, expressed in seconds since the first block. Block times must be monotonically increasing. Every channel must have at least one value inside a data block. The bdf-Toolbox provides functions to reasonably size the block length.

Timetable

The timetable at the end of the file specifies for each data block the elapsed time in seconds since the first data block and its position inside the file. It is used to quickly locate the data for a given time span in huge data files.

Supported Data Types

The bdf format provides several data types for storing data to a file. The data type can be individually selected for every channel. It should be considered that the data type has a significant impact on the size of the resulting file.

Unsigned Integer Data Types

BDF_BYTE_1

8 bit unsigned value, range 0..255

BDF_BYTE_2

16 bit unsigned value, range 0..65535

BDF_BYTE_3

24 bit unsigned value, range 0.. $2^{24} - 1$

BDF_BYTE_4

32 bit unsigned value, range 0.. $2^{32} - 1$

Signed Integer Data Types

BDF_INT_4

32 bit signed value

Floating Point Data Types

BDF_FLOAT_4

32 bit float value

BDF_DOUBLE_8

64 bit float value

Discrete Value Data Types

BDF_BIT_1

1 bit value

Note: Discrete values are packed into full bytes. Thus, the use of 1 to 8 discrete value channels has no impact on file size.

Data Compression

The toolbox functions allow a compression of the recorded data. The compression is performed individually for all data blocks. This still allows the fast access of data of a certain time without having to load larger parts of the file. On the other hand, the efficiency of the compression strongly depends on the structure and length of the defined data block. This means that e.g. for a very short data block containing multiple noisy high-frequency signals a very poor or even negative (due to overhead) compression ratio may result. Data blocks covering a longer time span and containing mostly static signals (e.g. bus values) deliver the best compression ratios. The definition of the data block structure should be carefully selected to achieve the best compression results.

Data compression consumes some processing time, so that it may be not appropriate for time critical applications.

Compression Methods

The compression methods are identified by the “CompressionID” value in the file header.

Standard compression (ID=1)

The standard compression uses the ‘zlib’ data compression functions that are provided by the ‘zlib’ general purpose compression library. For further information consult www.gzip.org.

Compressed data files differ from uncompressed ones in three major issues:

- The header value “CompressionID” is set to a value unequal to zero. This indicates the form of compression that is applied to the file.
- An additional header variable indicates the achieved overall compression ratio of the file.
- The lengths of the individual data block differ. The block header value “BlockLength” indicates the actual size in bytes for the given data block.

Note: To access data blocks in a compressed bdf-File, the timetable look-up must be used. A predictive approach by multiplying the number of blocks times the block size will fail!

Physical File Structure

This section describes the effective file structure as it is written to disk. The information given in this chapter should be used for reference only. Whenever applications shall read or create a bdf-file, the use of supplied routines from the appropriate bdf-toolbox is strongly recommended!

File Header

The file header for this release has a length of 160 (0x100) bytes and always starts at physical offset 0x0. The first three bytes must contain the file type identifier, which is 'BDF'.

<i>ofs</i>	<i>0</i>	<i>1</i>	<i>2</i>	<i>3</i>	<i>4</i>	<i>5</i>	<i>6</i>	<i>7</i>	<i>8</i>	<i>9</i>	<i>a</i>	<i>b</i>	<i>c</i>	<i>d</i>	<i>e</i>	<i>f</i>	
0	B	D	F		File Release ID				System ID				unused				
10	Data Start Time								Data End Time								
20	File Creation Time								UTC Offset								
30	Data Block Length								Compression ID				Realtime ID				
40	First Data Block Offset								No of Data Blocks				Data Block Size				
50	Timetable Offset								Timetable Size				Str	unused			
60	No of Header Vars				unused												
70	No of Channels				unused												
80	Calibration File				unused												
90	unused																

- **File Release Identifier**

The file release identifier occupies four bytes, starting at absolute byte offset 0x4:

File Offset (hex) →			
4	5	6	7
<i>uint32</i>			
File Release ID			

The release identifier is written as an unsigned integer value.

- **System Identifier**

The identifier of the file creating operating system is an unsigned integer number.

File Offset (hex) →			
c	d	e	f
<i>uint32</i>			
System ID			

- **Time Information**

The bytes from offset 0x10 to 0x27 contain three double numbers that represent time information values in serial format. Data Start Time, Data End Time and File Creation Time contain double values representing a serial UTC-time.

File Offset (hex) →															
10	11	12	13	14	15	16	17	18	19	1a	1b	1c	1d	1e	1f
<i>double</i>								<i>double</i>							
Data Start Time								Data End Time							

To convert the UTC-times into the local time of place of file creation, UTC Offset is added to the serial time. UTC Offset is a double value at offset 0x28, representing the local time zone in hours (e.g. Germany (winter) → UTC Offset=+1.0)

File Offset (hex) →															
20	21	22	23	24	25	26	27	28	29	2a	2b	2c	2d	2e	2f
<i>double</i>								<i>double</i>							
File Creation Time								UTC Offset							

• Data Block Information

Required information about the data section in the bdf-file is given in the bytes from offset 0x30 to 0x3f. The double value Data Block Length defines the time span covered by on data block in seconds.

File Offset (hex) →															
30	31	32	33	34	35	36	37	38	39	3a	3b	3c	3d	3e	3f
<i>double</i>								unused							
Data Block Length															

The identifier of the applied compression method is an unsigned integer number.

File Offset (hex) →			
38	39	3a	3b
<i>uint32</i>			
Compression ID			

First Data Block Offset is an unsigned, 8 byte long integer value that holds the file position of the first data block. The number of data blocks in the bdf-file is specified as an unsigned integer value, as the size of one (uncompressed) data block in bytes is.

File Offset (hex) →															
40	41	42	43	44	45	46	47	48	49	4a	4b	4c	4d	4e	4f
<i>uint64</i>								<i>uint32</i>				unused			
Data Block Offset								No of Data Blocks							

- **Timetable Information**

The location and the size of the timetable that is at the end of the bdf-file is given in bytes 0x50 to 0x5b. Timetable offset is the file position of the timetable, timetable size indicates the size of the timetable in bytes.

File Offset (hex) →															
50	51	52	53	54	55	56	57	58	59	5a	5b	5c	5d	5e	5f
<i>uint64</i>								<i>uint32</i>				unused			
Timetable Offset								Timetable Size							

- **Header Variables Information**

The number of user header variables is stored as unsigned integer from offset 0x60 on.

File Offset (hex) →															
60	61	62	63	64	65	66	67	68	69	6a	6b	6c	6d	6e	6f
<i>uint32</i>				unused											
Data Block Length															

• Channel Number Information

The number of channels in the bdf-file is stored as unsigned integer from offset 0x70 on.

File Offset (hex) →															
70	71	72	73	74	75	76	77	78	79	7a	7b	7c	7d	7e	7f
<i>uint32</i>				unused											
No of Channels															

• Calibration File

This flag indicates if an external calibration file must be used to interpret the data. The value is stored as unsigned integer, where “1” stands for the use of an external file.

File Offset (hex) →			
80	81	82	83
<i>uint32</i>			
Calibration File			

Header Variables

Following the file header at offset 0x100 is a section of user variables. The number *n* of variables can be set by the user and is specified in the header section. A variable is characterized by the name of the variable, a type specification and the value of a variable. The number of variables is unlimited.

File Header	Var 1	Var 2	Var 3	...	Var n	Channel Information
-------------	-------	-------	-------	-----	-------	---------------------

Name, type and value of the variables are stored as strings. Their content has no impact on the bdf-file itself, but may be important to applications that use the file, e.g. *testeval* requires some header variables to be defined and correctly set.

Relative File Offset (hex) →							
0	...	95	96	97	98	...	197
<i>char150</i>			<i>char2</i>		<i>char256</i>		
Name			Type		Value		

Channel Information

After the file header and its variables, the information about the stored channels is the next part of the bdf-file. This section consists of a basic channel header and user-defined channel variables for each channel. Both are written sequentially for all channels in the bdf-file. The number of channels in a bdf-file is not limited.

File Header	Header Variables	Channel 1 Header	Channel 1 Variables	...	Channel n Header	Channel n Variables	Data Section
--------------------	-------------------------	-------------------------	----------------------------	-----	-------------------------	----------------------------	---------------------

The channel header includes all the information necessary to read the channel variables and the channel data. It has the following format:

<i>ofs</i>	<i>0</i>	<i>1</i>	<i>2</i>	<i>3</i>	<i>4</i>	<i>5</i>	<i>6</i>	<i>7</i>	<i>8</i>	<i>9</i>	<i>a</i>	<i>b</i>	<i>c</i>	<i>d</i>	<i>e</i>	<i>f</i>
0	Channel Name															
...																
90						unused	Data Format			Data Block Offset						
a0	Occ per Block			BPV		S		No of ChannelVars			unused					
b0	Time Offset								unused							
c0	unused															
d0	unused															

- **Channel Name**

The Channel Name describes the mnemonics of the channel.

Relative File Offset (hex) →		
0	...	95
<i>char150</i>		
Channel Name		

- **Data Information**

The next section describes the data and the interpretation of the data that is stored for this channel.

<i>ofs</i>	<i>0</i>	<i>1</i>	<i>2</i>	<i>3</i>	<i>4</i>	<i>5</i>	<i>6</i>	<i>7</i>	<i>8</i>	<i>9</i>	<i>a</i>	<i>b</i>	<i>c</i>	<i>d</i>	<i>e</i>	<i>f</i>
90							unused	Data Format			Data Block Offset					
a0	Occ per Block			BPV												

Data Format specifies the type of stored data as an unsigned integer. Supported data formats with this release are 32-bit float numbers and 16-bit unsigned integers.

Data **Block Offset** is the number of bytes from the start of a data block to the first sample of this channel. This information is used to quickly access the channel data.

Occ. Per Block gives the number of samples for this channel inside one data block. It must be an integer number larger than zero.

BPV describes the Bytes Per Value used for the data of this channel. It is usually set by selecting the data format.

- **No of Channel Vars**

No of Channel Vars declares the number of user variables that follow the Channel Header. The number of user variables is not limited.

File Offset (hex) →			
a8	a9	aa	ab
<i>uint32</i>			
No of Channel Vars			

- **Time Offset**

Time Offset is a double number that describes the time offset of this channel compared to the Data Block Start time.

File Offset (hex) →							
b0	b1	b2	b3	b4	b5	b6	b7
<i>double</i>							
Time Offset							

- **Channel Variables**

The channel variables use the same format as the header variables do.

Data Section

The data section stores the values for all channels in a block-oriented manner. The number of data blocks within one file is limited to 2^{32} , whereas the length of one data block is limited to 2^{32} bytes. The maximum file size is therefore 2^{64} bytes or roughly $1,6 \cdot 10^5$ TByte.

File Header	Channel Information	Data Block 1	Data Block 2	...	Data Block 2	Time-table
-------------	---------------------	--------------	--------------	-----	--------------	------------

Every Data Block starts with a 16 byte long header. **Block Count** is the number of this block in the bdf-file, starting with 0.

Following is the **Block Start Time** as number of elapsed seconds since the Data Start Time declared in the file header.

Block Size declares the length of this data block in bytes. If no compression is used, the block size is identical to the one specified in the header.

Relative File Offset (hex) →															
0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
<i>uint32</i>				<i>double</i>								<i>uint32</i>			
Block Count				Block Start Time								Block Size			

After this header block, the values or samples of the channels are stored to disk. The samples are written in sequence for all channels. In the example shown below, channel 1 holds N samples, whereas channel 2 only has one sample per block. Every channel must have at least one sample per block.

Relative File Offset (hex) →											
S0	S1	...	Sn	S0			...	S0	S1	...	Sn
Channel 1				Channel 2				Channel n		

Timetable Section

The end of a bdf-file is marked by the timetable. The timetable is used to quickly access data at a certain time inside the file without having to read the entire file. The timetable is automatically generated when using the toolbox functions.

File Header	Channel Information	Data Section	t1	t2	...	t n
			Timetable			

The timetable holds information on starting time and file position for every one of the n data blocks in the current bdf-file.

Relative File Offset (hex) →															
0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
<i>double</i>								<i>uint64</i>							
Block Time								Block Position							

- **Block Time**

Number of elapsed seconds against the file start time.

- **Block Position**

Physical file position in bytes from the start of the file.

Index

BDF_BIT_1.....	9	Creating operating system.	
BDF_BYTE_1.....	8	15
BDF_BYTE_2.....	8	Data End Time.....	16
BDF_BYTE_3.....	8	Data section.....	5, 24
BDF_BYTE_4.....	8	Data Start Time.....	16
BDF_DOUBLE_8.....	8	Data type.....	7
BDF_FLOAT_4.....	8	Discrete values.....	9
BDF_INT_4.....	8	File Creation Time.....	16
Bdf-file release.....	1	File header.....	4, 14
Bdf-toolbox.....	13	File release identifier.....	15
BPV.....	22	File type identifier.....	14
Calibration file.....	19	Header variables.....	18
Compression.....	11	Testeval.....	5, 20
Compression methods...	12	Time base.....	3
		Timetable.....	5, 18
		UTC Offset.....	16