Product Registration

Thank you for purchasing YOKOGAWA products.

YOKOGAWA provides registered users with a variety of information and services. Please allow us to serve you best by completing the product registration form accessible from our homepage.

http://tmi.yokogawa.com/
Foreword
Thank you for purchasing the WVF File Access API (Model 707712).
This User’s Manual contains information about the installation, programming model, and functions of
the WE Control API. To ensure correct use, please read this manual thoroughly before operation.

Notes
• The contents of this manual describe the WVF File Access API Ver. 1.0.1. If you are using
another version of the API, the information given in this manual may differ from that of API
that you are using.
• The contents of this manual are subject to change without prior notice as a result of continuing
improvements to the instrument’s performance and functions.
• Every effort has been made in the preparation of this manual to ensure the accuracy of its contents.
However, should you have any questions or find any errors, please contact your nearest
YOKOGAWA dealer.
• Copying or reproducing any or all of the contents of this manual without YOKOGAWA’s permission is
strictly prohibited.

Trademarks
• Microsoft, Windows, and Windows NT are either registered trademarks or trademarks of Microsoft
Corporation in the United States and/or other countries.
• Adobe and Acrobat are trademarks of Adobe Systems Incorporated.
• All other company and product names used in this manual are trademarks or registered trademarks
of their respective companies.

Revisions
1st Edition: July 2003
Terms and Conditions of the Software License

NOTICE - PLEASE READ CAREFULLY BEFORE USE

Thank you very much for purchasing this medium containing a software program and related documentation provided by Yokogawa Electric Corporation (hereinafter called “Yokogawa”), and the program contained, embedded, inserted or used in the medium (hereinafter called the “Yokogawa Software Program”).

By opening this package or plastic wrapping (hereinafter called “Package”) enclosing the Yokogawa Software Program, you acknowledge that you understand and agree to the “Terms and Conditions of the Software License” (hereinafter called “Terms and Conditions”) which is written in the documentation and separately attached. Accordingly, the Terms and Conditions bind you.

Yokogawa hereby grants you permission to use the Yokogawa Software Program on the conditions that you agree to the Terms and Conditions before you open the Package and/or install it in or onto a computer.

IF YOU DO NOT AGREE TO THE TERMS AND CONDITIONS, YOU CANNOT OPEN THE PACKAGE, AND MUST IMMEDIATELY RETURN IT TO YOKOGAWA OR ITS DESIGNATED PARTY.

Terms and Conditions of the Software License

Yokogawa Electric Corporation, a Japanese corporation (hereinafter called “Yokogawa”), grants permission to use this Yokogawa Software Program (hereinafter called the “Licensed Software”) to the Licensee on the conditions that the Licensee agrees to the terms and conditions stipulated in Article 1 hereof.

You, as the Licensee (hereinafter called “Licensee”), shall agree to the following terms and conditions for the software license (hereinafter called “License”) based on the use intended for the Licensed Software.

Please note that Yokogawa grants the Licensee permission to use the Licensed Software under the terms and conditions herein and in no event shall Yokogawa intend to sell or transfer the Licensed Software to the Licensee.

Licensed Software Name: Model 707712 WVF File Access API

Number of License: 1

Article 1 (Scope Covered by these Terms and Conditions)

1.1 The terms and conditions stipulated herein shall be applied to any Licensee who purchases the Licensed Software on the condition that the Licensee consents to agree to the terms and conditions stipulated herein.

1.2 The “Licensed Software” herein shall mean and include all applicable programs and documentation, without limitation, all proprietary technology, algorithms, and know-how such as a factor, invariant or process contained therein.

Article 2 (Grant of License)

2.1 Yokogawa grants the Licensee, for the purpose of single use, non-exclusive and non-transferable license of the Licensed Software with the license fee separately agreed upon by both parties.

2.2 The Licensee is, unless otherwise agreed in writing by Yokogawa, not entitled to copy, change, sell, distribute, transfer, or sublicense the Licensed Software.

2.3 The Licensed Software shall not be copied in whole or in part except for keeping one (1) copy for back-up purposes. The Licensee shall secure or supervise the copy of the Licensed Software by the Licensee itself with great, strict, and due care.

2.4 In no event shall the Licensee dump, reverse assemble, reverse compile, or reverse engineer the Licensed Software so that the Licensee may translate the Licensed Software into other programs or change it into a man-readable form from the source code of the Licensed Software. Unless otherwise separately agreed by Yokogawa, Yokogawa shall not provide the Licensee the source code for the Licensed Software.

2.5 The Licensed Software and its related documentation shall be the proprietary property or trade secret of Yokogawa or a third party which grants Yokogawa the rights. In no event shall the Licensee be transferred, leased, sublicensed, or assigned any rights relating to the Licensed Software.

2.6 Yokogawa may use or add copy protection in or onto the Licensed Software. In no event shall the Licensee remove or attempt to remove such copy protection.

2.7 The Licensed Software may include a software program licensed for re-use by a third party (hereinafter called “Third Party Software”, which may include any software program from affiliates of Yokogawa made or coded by themselves.) In the case that Yokogawa is granted permission to sublicense to third parties by any licensors (sub-licensor) of the Third Party Software pursuant to different terms and conditions than those stipulated in this Agreement, the Licensee shall observe such terms and conditions of which Yokogawa notifies the Licensee in writing separately.

2.8 In no event shall the Licensee modify, remove or delete a copyright notice of Yokogawa and its licensor contained in the Licensed Software, including any copy thereof.

Article 3 (Restriction of Specific Use)

3.1 The Licensed Software shall not be intended specifically to be designed, developed, constructed, manufactured, distributed or maintained for the purpose of the following events:

a) Operation of any aviation, vessel, or support of those operations from the ground,

b) Operation of nuclear products and/or facilities,

c) Operation of nuclear weapons and/or chemical weapons and/or biological weapons; or

d) Operation of medical instrumentation directly utilized for humankind or the human body.

3.2 Even if the Licensee uses the Licensed Software for the purposes in the preceding Paragraph 3.1, Yokogawa has no liability to or responsibility for any demand or damage arising out of the use or operations of the Licensed Software, and the Licensee agrees, on its own responsibility, to solve and settle the claims and damages and to defend, indemnify or hold Yokogawa totally harmless, from or against any liabilities, losses, damages and expenses (including fees for recalling the Products and reasonable attorney’s fees and court costs), or claims arising out of and related to the above-said claims and damages.

Article 4 (Warranty)

4.1 The Licensee shall agree that the Licensed Software shall be provided to the Licensee on an “as is” basis when delivered. If defect(s), such as damage to the medium of the Licensed Software, attributable to Yokogawa is found, Yokogawa in its own judgement that it is necessary, Yokogawa may from time to time provide the Licensee with Revision upgrades and Version upgrades separately specified by Yokogawa (hereinafter called “Updates”).

4.2 When Yokogawa decides in its own judgement that it is necessary, Yokogawa may from time to time provide the Licensee with Revision upgrades and Version upgrades separately specified by Yokogawa (hereinafter called “Updates”).

4.3 Notwithstanding the preceding Paragraph 4.1, when third party software is included in the Licensed Software, the warranty period and terms and conditions that apply shall be those established by the provider of the third party software.
4.4 Notwithstanding the preceding Paragraph 4.3, in no event shall Yokogawa provide Updates where the Licensee or any third party conducted renovation or improvement of the Licensed Software.

4.5 THE FOREGOING WARRANTIES ARE EXCLUSIVE AND IN LIEU OF ALL OTHER WARRANTIES OF QUALITY AND PERFORMANCE, WRITTEN, ORAL, OR IMPLIED, AND ALL OTHER WARRANTIES INCLUDING ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE ARE HEREBY DISCLAIMED BY YOKOGAWA AND ALL THIRD PARTIES LICENSING THIRD PARTY SOFTWARE TO YOKOGAWA.

4.6 Correction of nonconformity in the manner and for the period of time provided above shall be the Licensee’s sole and exclusive remedy for any failure of Yokogawa to comply with its obligations and shall constitute fulfillment of all liabilities of Yokogawa and any third party licensing the Third Party Software to Yokogawa (including any liability for direct, indirect, special, incidental or consequential damages) whether in warranty, contract, tort (including negligence but excluding willful conduct or gross negligence by Yokogawa) or otherwise with respect to or arising out of the use of the Licensed Software.

Article 5 (Infringement)

5.1 If and when any third party should demand injunction, initiate a law suit, or demand compensation for damages against the Licensee under patent right (including utility model right, design patent, and trade mark), copy right, and any other rights relating to any of the Licensed Software, the Licensee shall promptly notify Yokogawa in writing to that effect without delay.

5.2 In the case of the preceding Paragraph 5.1, the Licensee shall assign to Yokogawa all of the rights to defend the Licensee and to negotiate with the claiming party. Furthermore, the Licensee shall provide Yokogawa with necessary information or any other assistance for Yokogawa’s defense and negotiation. If and when such a claim should be attributable to Yokogawa, subject to the written notice to Yokogawa stated in the preceding Paragraph 5.1, Yokogawa shall defend the Licensee and negotiate with the claiming party at Yokogawa’s cost and expense and be responsible for the final settlement or judgment granted to the claiming party in the preceding Paragraph 5.1.

5.3 When any assertion or allegation of the infringement of the third party’s rights defined in Paragraph 5.1 is made, or when at Yokogawa’s judgment there is possibility of such assertion or allegation, Yokogawa will, at its own discretion, take any of the following countermeasures at Yokogawa’s cost and expense.

a) To acquire the necessary right from a third party which has lawful ownership of the right so that the Licensee will be able to continue to use the Licensed Software;

b) To replace the Licensed Software with an alternative one which avoids the infringement; or

c) To remodel the Licensed Software so that the Licensed Software can avoid the infringement of such third party’s right.

5.4 If and when Yokogawa fails to take either of the countermeasures as set forth in the preceding subparagraphs of Paragraph 5.3, Yokogawa shall indemnify the Licensee only by paying back the price amount of the Licensed Software which Yokogawa has received from the Licensee. THE FOREGOING PARAGRAPHS STATE THE ENTIRE LIABILITY OF YOKOGAWA AND ANY THIRD PARTY LICENSING THIRD PARTY SOFTWARE TO YOKOGAWA WITH RESPECT TO INFRINGEMENT OF THE INTELLECTUAL PROPERTY RIGHTS INCLUDING BUT NOT LIMITED TO, PATENT AND COPYRIGHT.

Article 6 (Liabilities)

6.1 If and when the Licensee should incur any damage relating to or arising out of the Licensed Software or service that Yokogawa has provided to the Licensee under the conditions herein due to a reason attributable to Yokogawa, Yokogawa shall take actions in accordance with this Agreement. However, in no event shall Yokogawa be liable or responsible for any special, incidental, consequential and/or indirect damage, whether in contract, warranty, tort, negligence, strict liability, or otherwise, including, without limitation, loss of operational profit or revenue, loss of use of the Licensed Software, or any associated products or equipment, cost of capital, loss or cost of interruption of the Licensee’s business, substitute equipment, facilities or services, offtime costs, delays, and loss of business information, or claims of customers of Licensee or other third parties for such or other damages. Even if Yokogawa is liable or responsible for the damages attributable to Yokogawa and to the extent of Article 6, Yokogawa’s liability for the Licensee’s damage shall not exceed the price amount of the Licensed Software or service fee which Yokogawa has received. Please note that Yokogawa shall be released or discharged from part or all of the liability under this Agreement if the Licensee modifies, remodels, combines with other software or products, or causes any deviation from the basic specifications or functional specifications, without Yokogawa’s prior written consent.

6.2 All causes of action against Yokogawa arising out of or relating to this Agreement or the performance or breach hereof shall expire unless Yokogawa is notified of the claim within one (1) year of its occurrence.

6.3 In no event, regardless of cause, shall Yokogawa assume responsibility for or be liable for penalties or penalty clauses in any contracts between the Licensee and its customers.

Article 7 (Limit of Export)

Unless otherwise agreed by Yokogawa, the Licensee shall not directly or indirectly export or transfer the Licensed Software to any countries other than those where Yokogawa permits export in advance.

Article 8 (Term)

This Agreement shall become effective on the date when the Licensee receives the Licensed Software and continues in effect unless or until terminated as provided herein, or the Licensee ceases using the Licensed Software by itself or with Yokogawa’s thirty (30) days prior written notice to the Licensee.

Article 9 (Injunction for Use)

During the term of this Agreement, Yokogawa may, at its own discretion, demand injunction against the Licensee in case that Yokogawa deems that the Licensed Software is used improperly or under severer environments other than those where Yokogawa has first approved, or any other condition which Yokogawa may not permit.

Article 10 (Termination)

Yokogawa, at its sole discretion, may terminate this Agreement without any notice or reminder to the Licensee if the Licensee violates or fails to perform this Agreement. However, Articles 5, 6, and 11 shall survive even after the termination.

Article 11 (Jurisdiction)

Any dispute, controversy, or differences between the parties hereto as to interpretation or execution of this Agreement shall be resolved amicably through negotiation between the parties upon the basis of mutual trust. Should the parties fail to agree within ninety (90) days after notice from one of the parties to the other, both parties hereby irrevocably submit to the exclusive jurisdiction of the Tokyo District Court (main office) in Japan for settlement of the dispute.

Article 12 (Governing Law)

This Agreement shall be governed by and construed in accordance with the laws of Japan. The Licensee expressly agrees to waive absolutely and irrevocably and to the fullest extent permissible under applicable law any rights against the laws of Japan which it may have pursuant to the Licensee’s local law.

Article 13 (Severability)

In the event that any provision hereof is declared or found to be illegal by any court or tribunal of competent jurisdiction, such provision shall be null and void with respect to the jurisdiction of that court or tribunal and all the remaining provisions hereof shall remain in full force and effect.
Contents

Foreword ................................................................................................................................................................. 1
Terms and Conditions of the Software License .................................................................................................. 2

1. Overview ......................................................................................................................................................... 1-1
2. Installation ...................................................................................................................................................... 2-1

3. File Operation Functions ................................................................................................................................ 3-1
   3.1 Basic Model ................................................................................................................................................. 3-1
   3.2 File Format .................................................................................................................................................. 3-1
   3.3 Model When Performing Write Operation of Measured Data ............................................................ 3-2
   3.4 Model When Performing Read Operation of Measured Data ............................................................ 3-4
   3.5 Common Information Structure and Channel Information Structure .............................................. 3-8
   3.6 The List of Functions ................................................................................................................................. 3-10
   3.7 Single File Access .................................................................................................................................. 3-11
      WeDPHeaderReadS ................................................................................................................................... 3-11
      WeDPDataRead ......................................................................................................................................... 3-11
      WeDPHeaderWriteS .................................................................................................................................. 3-12
      WeDPDataWrite ....................................................................................................................................... 3-13
   3.8 Sequential File Access .............................................................................................................................. 3-15
      WeDPHeaderCsReadS ............................................................................................................................... 3-15
      WeDPNsCsRead ......................................................................................................................................... 3-15
      WeDPHeaderCsWriteS ............................................................................................................................... 3-16
      WeDPNsCsWrite ....................................................................................................................................... 3-17
   3.9 Access the Specified Item of the Header File ................................................................................ .. 3-19
      WeDPHeaderItemRead ............................................................................................................................... 3-19
      WeDPHeaderItemWrite ............................................................................................................................... 3-19
   3.10 Data Operation ....................................................................................................................................... 3-21
      WeDPGetSampleChNum ............................................................................................................................ 3-21
      WeDPGetBlockNum .................................................................................................................................. 3-21
      WeDPInitializeAcqInfo .............................................................................................................................. 3-22
   3.11 The List of C Language Interface ........................................................................................... ........... 3-24

4. Error Cord ........................................................................................................................................................ 4-1

Index ..............................................................................................................................................................Index-1
1. Overview

This user’s manual describes the interface functions (WVF File Access API) used to access the waveform data files (WVF files).

The WVF File Access API is a dynamic link library (DLL) for Microsoft Windows. The DLL can be used in Win32 program development platforms such as Microsoft Visual C++ and Microsoft Visual Basic.

A List of Files Included

<table>
<thead>
<tr>
<th>File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WeAscii.bas</td>
<td>File containing definitions of constants for Visual Basic</td>
</tr>
<tr>
<td>WeDP.bas</td>
<td>File access function API definition file for Visual Basic</td>
</tr>
<tr>
<td>*.dll</td>
<td>DLL files for the WVF File Access API</td>
</tr>
<tr>
<td>WeAscii.h</td>
<td>File containing definitions of constants for Visual C++</td>
</tr>
<tr>
<td>WeDP.h</td>
<td>File access function API definition file for Visual C++</td>
</tr>
<tr>
<td>WeDP.pdf</td>
<td>WVF File Access Online Manual</td>
</tr>
<tr>
<td></td>
<td>(Adobe Acrobat Reader 3.0 or later required for opening the file.)</td>
</tr>
<tr>
<td>WeFileAccess</td>
<td>DLL file for the .Net development platform</td>
</tr>
</tbody>
</table>

The definition files above contain definitions of symbols, data structures, and other elements that are used by the API. Be sure to include them in the user application programs.

By default, the files listed above excluding the DLL files are copied to the C:\Program Files\WE7000\API directory. The DLL files are copied to directories indicated below in addition to the directory indicated above.

On Windows 95/98/Me: Windows\System
On Windows NT 4.0, 2000 Pro, XP Professional/Home Edition: Windows\System32

Supported OSs
Microsoft Windows 95/98/Me, Windows NT 4.0, Windows 2000 Pro or Windows XP Professional/Home Edition

Supported Development Platforms
Microsoft Visual Basic 5.0, 6.0, and .Net
Microsoft Visual C++6.0 and .Net
Microsoft Visual C#.Net

Applicable Models
WE7000
DL708, DL708E, DL716, and DL750
DL1720 and DL1740
DL1600 Series
DL1500 Series
DL7100 and DL7200
DL7400 Series

Note
Class names and function names of the .Net-compatible class library
- The namespace is WeFileAccess. The class name is WeFile.
- The names of the .Net functions are obtained by removing “WeDP” from the “WeDPxxx” function names.
2. Installation

This chapter describes the installation procedure for the “WVF File Access API.”

1. Insert the WVF File Access API Setup Disk into your PC’s CD-ROM drive. An installer automatically starts and the following dialog box opens. Click **Next**.

2. The following dialog box opens containing license agreement information. Confirm the license agreement, click the I accept the terms in the license agreement option button, and click **Next**.
3. Specify the installation destination. The default installation destination is set to “C:\Program Files\WE7000\API\.” If this is OK, click Next. To change the installation directory, click Change, select the directory, and click Next.

4. A dialog box appears confirming that you wish to begin installation. Click Install.
5. The installation starts and the following dialog box opens indicating the progress of the installation.

6. A dialog box opens notifying you that the installation has been completed. Click Finish.
3. File Operation Functions

3.1 Basic Model

For the write operation of measured data, functions are provided for the following two assumed models: one in which data blocks are added in order to a single file (Single File Model) and another in which a file is created for each data block (Sequential File Model).

For the read operation of measured data, functions are provided for the following two assumed models: one in which data is read by specifying a block from a file in which data is stored in multiple blocks (Single File Model) and another in which data is read by specifying the number of samples from multiple files (Sequential File Model).

3.2 File Format

The files are in YOKOGAWA's proprietary format (the same format as the measured data that is saved in binary format using the WE7000 Control Software (.wvf file)). There are three types of storage formats of .wvf files: block type, trace type, and scan type. This API can be used to read the measured data without being aware of the storage format of .wvf files. However, when reading a Scan type file, the entire data is assumed to be a single block. This API stores the data using the block type of the .wvf format.

*Note*

For details on .wvf files, see technical information (TI7000-21E) issued by YOKOGAWA.
3.3 Model When Performing Write Operation of Measured Data

Single File Model for Data Created Using an Application Program (Multiple blocks are stored to the same file)

1. **Data processing in the memory**
2. **Data information structure**
3. **Data buffer**
4. **Scale data buffer**

- **Store the required data in the data information structure.**
  - `WEDPInitializeAcqInfo`
- **Write the header file of the single file.**
  - `WedPHeaderWriteS`
- **Write the data file of the single file.**
  - `WedPDataWrite`
Creating Sequential Files (Store data that has been created using a program to a single file for each block)

- **Data file**
- **Header file**
- **Scale data buffer**
- **Data buffer**
- **Data information structure**
- **Data processing in the memory**

Store the required data in the data information structure. 
**WEDPInitializeAcqInfo**

Write the header file of the sequential file. 
**WeDPHeaderCsWrite**

Write the data file of the sequential file. 
**WeDPCsWrite**
3.4 Model When Performing Read Operation of Measured Data

Single File Model (Read data by specifying blocks)

WeDPDataRead(FileName,BlockNo,ChNo,DataForm,DataBuff)

FileName: File name (excluding the sequential number section).
BlockNo: Block number (0 to n).
ChNo: Channel number. Specify -1 to read all channels.
DataForm: Data type.
DataBuf: Pointer to the buffer in which the data is to be stored.

If you specify WeDPDataRead("FileName",1,-1,DATA_SINGLE,DataBuff), the data in the shaded section in the following figure is stored to DataBuff.
Sequential File Model (Read data by specifying the number of samples)

![Diagram of Sequential File Model]

**WeDpCsRead**(*FileName*, *SeriesNo*, *Start*, *Length*, *ChNo*, *DataForm*, *DataBuff*)

- **FileName**: File name (excluding the sequential number section).
- **SeriesNo**: First sequence number.
- **Start**: Start sample number.
- **Length**: Number of data points per channel to be read.
- **ChNo**: Channel number. Specify -1 to read all channels.
- **DataForm**: DataBuff type.
- **DataBuff**: Pointer to the buffer in which the data is to be stored.

If you specify **WeDpCsRead**("FileName", 1, 200, 500, -1, DATA_SINGLE, DataBuff), the data in the shaded section in the following figure is stored to DataBuff.

```
<table>
<thead>
<tr>
<th>.FileName00000</th>
<th>FileName00001</th>
<th>FileName00002</th>
<th>FileName00003</th>
<th>FileNamennnnn</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>200</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>299</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

This file has "4 channels \(\times\) 300 samples" of data stored.

Data is stored to the memory after scaling with VResolution and VOffset in the .hdr file.
Search Method of Sequential Files

- When the sequence number is set to -1
  
  \[ \text{WeDPCsRead("FileName",-1,200,500,-1,WE\_FLOAT,DataBuff)} \]

  Names of the files created are:
  
  "FileName.hdr" • "FileName.wvf"

- When the sequence number is set to a positive integer
  
  \[ \text{WeDPCsRead("FileName",2,200,500,-1,WE\_FLOAT,DataBuff)} \]

  Name of the .wvf file created
  "FileName00002.wvf"  
  Does not exist

  Name of the header file created
  "FileName00002.hdr"  
  Exist

  Names of the files created are:
  
  "FileName00002.hdr" • "FileName00002.wvf"

  Name of the header file created
  "FileName.hdr"  
  Does not exist

  Name of the .wvf file created
  "FileName00002.wvf"  
  Exist

  Names of the files created are:
  
  "FileName.hdr" • "FileName00002.wvf"

  Error
Reading Header Files

- Name of the header file for single files

```c
WeDPHeaderReadS(FileName, BlockNo, ComBuff(), ChBuff())
```

```c
WeDPHeaderReadS("FileName", 2, ComBuff(), ChBuff())
```

FileName: "FileName"
BlockNo: 2
ComBuff(): ComBuff()
ChBuff(): When ChBuff()

Set to FileName.hdr.

- Name of the header file for sequential files

```c
WeDPHeaderCsReadS(FileName, SeriesNo, ComBuff(), ChBuff())
```

```c
WeDPHeaderCsReadS("FileName", 2, ComBuff(), ChBuff())
```

FileName: "FileName"
SeriesNo: 2
ComBuff(): ComBuff()
ChBuff(): When ChBuff()

Set to FileName00002.hdr.

```c
WeDPHeaderCsReadS("FileName", -1, ComBuff(), ChBuff())
```

FileName: "FileName"
SeriesNo: -1
ComBuff(): ComBuff()
ChBuff(): When ChBuff()

Set to FileName.hdr.
3.5 Common Information Structure and Channel Information Structure

User-Defined Type
When the X-axis data is the same on all channels, the data structure is as follows:

- Common information
  
  Type CommonInf1
  
  Comment As String*256 // Comment (set up to 255 bytes)
  SampleNum As Long // Number of samples
  ChNum As Long // Number of channels
  SamplingInterval As Double // Sampling interval
  PreTrigger As Long // Amount of pretrigger
  XUnit As String*16 // X-axis unit
  Date As String*16 // Measurement date (set up to 15 bytes)
  Time As String*16 // Measurement time (set up to 15 bytes)
  End Type

- Channel information
  
  Type ChanelInf1
  
  ChName AS String*16 // Channel name (set up to 15 bytes)
  ScaleA As Double // Scaling coefficient A
  ScaleB As Double // Scaling coefficient B
  Unit As String*16 // Unit (set up to 15 bytes)
  End Type

Note
The current version does not allow handling of files in which the X-axis data is not the same for all channels.

Correspondence between User-Defined Type and Header File Items

<table>
<thead>
<tr>
<th>User-Defined Type</th>
<th>Header File Item</th>
</tr>
</thead>
<tbody>
<tr>
<td>Comment</td>
<td>Character string in the comment line</td>
</tr>
<tr>
<td>SampleNum</td>
<td>Sum of BlockSize for the number of blocks (information of the first channel)</td>
</tr>
<tr>
<td>ChNum</td>
<td>TraceTotalNumber value</td>
</tr>
<tr>
<td>SamplingInterval</td>
<td>HResolution value (information of the first channel)</td>
</tr>
<tr>
<td>PreTrigger</td>
<td>HOffset value (information of the first channel)</td>
</tr>
<tr>
<td>XUnit</td>
<td>HUnit value (information of the first channel)</td>
</tr>
<tr>
<td>Date</td>
<td>Date value (information of the first channel)</td>
</tr>
<tr>
<td>Time</td>
<td>Time value (information of the first channel)</td>
</tr>
<tr>
<td>ChName</td>
<td>TraceName value (for each channel)</td>
</tr>
<tr>
<td>ScaleA</td>
<td>Linear VResolution value (for each channel)</td>
</tr>
<tr>
<td>ScaleB</td>
<td>Linear VOffset value (for each channel)</td>
</tr>
<tr>
<td>Unit</td>
<td>Linear Unit value (for each channel)</td>
</tr>
</tbody>
</table>
Correspondence between Acquisition Data Information and Header File Items

The correspondence between acquisition items (acquisition data information that can be read using WeGetAcqDataInfo (page 6-66)) and the header file items is as follows:

<table>
<thead>
<tr>
<th>Acquisition Data Information Item</th>
<th>Header File Item</th>
</tr>
</thead>
<tbody>
<tr>
<td>Comment</td>
<td>Character string in the comment line</td>
</tr>
<tr>
<td>Channel</td>
<td>–</td>
</tr>
<tr>
<td>DataType</td>
<td>VdataType</td>
</tr>
<tr>
<td>BlockNum</td>
<td>BlockNumber</td>
</tr>
<tr>
<td>StartBit</td>
<td>–</td>
</tr>
<tr>
<td>EffectiveBit</td>
<td>–</td>
</tr>
<tr>
<td>TrigActive</td>
<td>–</td>
</tr>
<tr>
<td>Record</td>
<td>BlockSize</td>
</tr>
<tr>
<td>Recordlen</td>
<td>–</td>
</tr>
<tr>
<td>TrigPosition</td>
<td>HOffset</td>
</tr>
<tr>
<td>Interval</td>
<td>HResolution</td>
</tr>
<tr>
<td>VRResolution</td>
<td>VRResolution</td>
</tr>
<tr>
<td>VOffset</td>
<td>VOffset</td>
</tr>
<tr>
<td>TrigLevel</td>
<td>–</td>
</tr>
<tr>
<td>TrigWidth</td>
<td>–</td>
</tr>
<tr>
<td>PlusOverData</td>
<td>VPlusOverData</td>
</tr>
<tr>
<td>MinusOverData</td>
<td>VMinusOverData</td>
</tr>
<tr>
<td>NonData</td>
<td>VIllegalData</td>
</tr>
<tr>
<td>DispMaxData</td>
<td>VMaxData</td>
</tr>
<tr>
<td>DispMinData</td>
<td>VMinData</td>
</tr>
</tbody>
</table>

The settings of the items of the header file that is created when writing measured data using this API are indicated in the table below. For a description of the header file of other applicable models, see the user’s manual that came with the product.

<table>
<thead>
<tr>
<th>Item Name</th>
<th>Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>FormatVersion</td>
<td>1.01</td>
</tr>
<tr>
<td>Model</td>
<td>WE7000</td>
</tr>
<tr>
<td>Endian</td>
<td>Lttl</td>
</tr>
<tr>
<td>DataFormat</td>
<td>Block</td>
</tr>
<tr>
<td>GroupNumber</td>
<td>Automatically created to match the stored data when writing measured data</td>
</tr>
<tr>
<td>TraceTotalNumber</td>
<td>Automatically created to match the stored data when writing measured data</td>
</tr>
<tr>
<td>DataOffset</td>
<td>0</td>
</tr>
<tr>
<td>TraceNumber</td>
<td>Automatically created to match the stored data when writing measured data</td>
</tr>
<tr>
<td>BlockNumber</td>
<td>Automatically created to match the stored data when writing measured data</td>
</tr>
<tr>
<td>TraceName</td>
<td>Set to Ch1 to ChN to match the stored data when writing measured data</td>
</tr>
<tr>
<td>BlockSize</td>
<td>Record of acquisition data information</td>
</tr>
<tr>
<td>VRResolution</td>
<td>VRResolution of acquisition data information</td>
</tr>
<tr>
<td>VOffset</td>
<td>VOffset of acquisition data information</td>
</tr>
<tr>
<td>VdataType</td>
<td>Set from parameters when writing measured data</td>
</tr>
<tr>
<td>VUnit</td>
<td>–</td>
</tr>
<tr>
<td>VPlusOverData</td>
<td>PlusOverData of acquisition data information</td>
</tr>
<tr>
<td>VminusOverData</td>
<td>MinusOverData of acquisition data information</td>
</tr>
<tr>
<td>VIllegalData</td>
<td>NonData of acquisition data information</td>
</tr>
<tr>
<td>VMaxData</td>
<td>DispMaxData of acquisition data information</td>
</tr>
<tr>
<td>VMinData</td>
<td>DispMinData of acquisition data information</td>
</tr>
<tr>
<td>HResolution</td>
<td>SamplingInterval of CommonInf1</td>
</tr>
<tr>
<td>HOffset</td>
<td>PreTrigger of CommonInf1</td>
</tr>
<tr>
<td>HUnit</td>
<td>XUnit of CommonInf1</td>
</tr>
<tr>
<td>Date</td>
<td>Date of CommonInf1</td>
</tr>
<tr>
<td>Time</td>
<td>Time of CommonInf1</td>
</tr>
<tr>
<td>PLinearSlope</td>
<td>ScaleA of CommonInf1</td>
</tr>
<tr>
<td>PLinearIntercept</td>
<td>ScaleB of CommonInf1</td>
</tr>
<tr>
<td>PLinearUnit</td>
<td>Unit of CommonInf1</td>
</tr>
<tr>
<td>PTraceName</td>
<td>ChannelName of CommonInf1</td>
</tr>
<tr>
<td>PLinearMode</td>
<td>“OFF”</td>
</tr>
</tbody>
</table>
### 3.6 The List of Functions

#### Single File Access

<table>
<thead>
<tr>
<th>Function Name</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>WeDPHeaderReadS</td>
<td>Read the header file of the single file.</td>
<td>3-11</td>
</tr>
<tr>
<td>WeDPDataRead</td>
<td>Read the data file of the single file.</td>
<td>3-11</td>
</tr>
<tr>
<td>WeDPHeaderWriteS</td>
<td>Write the header file of the single file.</td>
<td>3-12</td>
</tr>
<tr>
<td>WeDPDataWrite</td>
<td>Write the data file of the single file.</td>
<td>3-13</td>
</tr>
</tbody>
</table>

#### Sequential File Access

<table>
<thead>
<tr>
<th>Function Name</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>WeDPHeaderCsReadS</td>
<td>Read the header file of the sequential file.</td>
<td>3-15</td>
</tr>
<tr>
<td>WeDPCsRead</td>
<td>Read the data file of the sequential file.</td>
<td>3-15</td>
</tr>
<tr>
<td>WeDPHeaderCsWriteS</td>
<td>Write the header file of the sequential file.</td>
<td>3-16</td>
</tr>
<tr>
<td>WeDPCsWrite</td>
<td>Write the data file of the sequential file.</td>
<td>3-17</td>
</tr>
</tbody>
</table>

#### Access the Specified Item of the Header File

<table>
<thead>
<tr>
<th>Function Name</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>WeDPHeaderItemRead</td>
<td>Read the data of the specified item.</td>
<td>3-19</td>
</tr>
<tr>
<td>WeDPHeaderItemWrite</td>
<td>Write the data of the specified item.</td>
<td>3-19</td>
</tr>
</tbody>
</table>

#### Data Operation

<table>
<thead>
<tr>
<th>Function Name</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>WeDPGetSampleChNum</td>
<td>Get the number of samples and number of channels.</td>
<td>3-21</td>
</tr>
<tr>
<td>WeDPGetBlockNum</td>
<td>Get the number of blocks.</td>
<td>3-21</td>
</tr>
<tr>
<td>WeDPInitializeAcqInfo</td>
<td>Store the required data in the data information structure.</td>
<td>3-22</td>
</tr>
</tbody>
</table>
3.7 Single File Access

WeDPHeaderReadS

Description
Reads the data from the header file by specifying the block number.

Syntax
WeDPHeaderReadS(ByVal FileName As String, ByVal BlockNo As Long, ByRef ComBuff() As CommonInf1, ByRef ChBuff() As ChanelInf1)

Return value
Returns 0 if successful. Returns an error code if unsuccessful.

Parameters
FileName (IN) Header file name.
BlockNo (IN) Block number. Specify the block number you wish to read.
ComBuff() (OUT) Common information structure. Define using an array.
ChBuff() (OUT) Channel information structure. Define using an array.

Note:
Collectively retrieves the header file information for data of which the number of X-axis data points is the same. Prepare a single-element array for the common information structure buffer. Prepare channel information structure buffer for the amount equal to the number of channels.

Example (Visual Basic)
Dim ret As Long
Dim FileName As String
Dim BlockNo As Long
Dim ComBuff(0) As CommonInf1
Dim ChBuff(5) As ChanelInf1
FileName = “TestData1”
BlockNo = 2
ret = WeDPHeaderReadS(FileName, BlockNo, ComBuff(), ChBuff())
If 0 <> ret Then
   MsgBox “File read error”
End If

Note
This function collectively retrieves a section of the data in the header file as indicated in section 3.5, “Common Information Structure and Channel Information Structure.” To read unsupported items, use WeDPHeaderItemRead (page 3-19).

WeDPDataRead

Description
Reads the data from the data file by specifying the block number.

Syntax
WeDPDataRead(ByVal FileName As String, ByVal BlockNo As Long, ByVal ChNo As Long, ByVal DataForm As Long, ByRef DataBuff As Any)
Return value
Returns 0 if successful. Returns an error code if unsuccessful.

Parameters
- **FileName** (IN) Data file name. Specify the file name excluding the extension.
- **BlockNo** (IN) Block number.
- **ChNo** (IN) Channel number. -1 represents all channels. Counted from 1.
  If you specify -1, pass a buffer defined using a two-dimensional array for DataBuff.
- **DataForm** (IN) DataBuff type. Specify BYTE (char), Integer (short), Long (long), Single (float), or Double (double). No other data types can be passed. The type inside the parentheses indicates the type for the C Language.
- **DataBuff** (OUT) Data storage buffer. Pass the buffer in the type specified by DataForm.

**Note:**
Reads the data from the data file in units of blocks. The files cannot be handled as sequential files. Specify DataForm from the following defined data.
- **BYTE:** WE_UBYTE
- **Integer:** WE_SWORD
- **Long:** WE_SLONG
- **Single:** WE_FLOAT
- **Double:** WE_DOUBLE

**Example (Visual Basic)**

```vbnet
Dim FileName As String
Dim BlockNo As Long
Dim ChNo As Long
Dim DataForm As Long
Dim DataBuff(999,3) As Single
FileName = “TestData1”
BlockNo = 2
ChNo = -1
DataForm = WE_FLOAT
ret = WeDPDataRead(FileName,BlockNo,ChNo,DataForm,DataBuff(0,0))
If 0 <> ret Then
   MsgBox “File read error”
End If
```

**WeDPHeaderWriteS**

**Description**
Writes the header information at once to the header file by specifying the block.

**Syntax**

```
WeDPHeaderWriteS(ByVal FileName As String, ByVal BlockNo As Long, ByRef ComBuff() As CommonInf1, ByRef ChBuff() As ChanelInf1, ByRef AcqInfo As AcqDataInfEx)
```

**Return value**
Returns 0 if successful. Returns an error code if unsuccessful.

**Parameters**
- **FileName** (IN) Header file name.
- **BlockNo** (IN) Block number. Specify the block number to be written.
- **ComBuff()** (IN) Common information structure. Define using an array.
- **ChBuff()** (IN) Channel information structure. Define using an array.
- **AcqInfo** (IN) Data information structure (AcqDataInfoEx).
**Note:**
Collectively writes the header file information for data of which the number of X-axis data points is the same. Prepare a single-element array for the common information structure buffer. Prepare channel information structure buffer for the amount equal to the number of channels. AcqInfo passes the data retrieved using WeAcqInfoInitialize(). Or, create the data by referring to section 3.5, “Common Information Structure and Channel Information Structure.” Data is written to the specified block.

**Example (Visual Basic)**
```
Dim ret As Long
Dim FileName As String
Dim BlockNo As Long
Dim CommonBuff(0) As CommonInf1
Dim ChBuff(5) As ChanelInf1
Dim AcqInfo(5) As AcqDataInfoEx
Dim InfoNum As Long
BlockNo=2
Ret = WeGetAcqDataInfoEx(hMo,-1,BlockNo,AcqInfo(),infoNum)
If 0 <> ret Then
    MsgBox “Data information read error”
EndIf
FileName = “TestData1”
ret=WeDPHeaderWriteS(FileName,BlockNo,ComBuff(),ChBuff(),AcqInfo())
If 0 <> ret Then
    MsgBox “File write error”
End If
```

**WeDPDataWrite**
**Description**
Writes the data to the data file in units of blocks.

**Syntax**
```
WeDPDataWrite(ByVal FileName As String,ByVal BlockNo As Long,ByVal SampleNum As Long,ByRef AcqInfo As AcqDataInfoEx,ByVal DataForm As Long,ByRef DataBuff As Any)
```

**Return value**
Returns 0 if successful. Returns an error code if unsuccessful.

**Parameters**
- **FileName (IN)**: Data file name. Specify the file name excluding the extension. A .wvf extension is added to the created file.
- **BlockNo (IN)**: Block number.
- **SampleNum (IN)**: Number of samples.
- **AcqInfo (IN)**: Data information structure. AcqInfo passes the data retrieved using WeAcqInfoInitialize(). Or, create the data by referring to section 3.5, “Common Information Structure and Channel Information Structure.”
- **DataForm (IN)**: DataBuff type. Specify BYTE (char), Integer (short), Long (long), Single (float), or Double (double). No other data types can be passed. The type inside the parentheses indicates the type for the C Language.
- **DataBuff (OUT)**: Data storage buffer. Pass the buffer in the type specified by DataForm. Pass a buffer defined using a two-dimensional array.
**Note:**
Write data to a data file. The files cannot be handled as sequential files. Specify DataForm from the following defined data.
- **BYTE:** WE_UBYTE
- **Integer:** WE_SWORD
- **Long:** WE_SLONG
- **Single:** WE_FLOAT
- **Double:** WE_DOUBLE

**Example (Visual Basic)**
```vbnet
dim filename as string
Dim blockno as long
Dim sampleNum as long
Dim acqInfo() as acqDataInfoEx
Dim dataForm as long
Dim dataBuff(999,1) as single
Dim infoNum as integer
BlockNo = 2
SampleNum = 1000
WeGetAcqDataInfoEx(hMo,-1,BlockNo,AcqInfo(),InfoNum)
filename = “TestData1”
DataForm = WE_FLOAT
ret = WeDPDataWrite(filename,BlockNo,SampleNum,AcqInfo(),DataForm,
dataBuff(0,0))
If 0 <> ret Then
    MsgBox “File write error”
End If
```
### 3.8 Sequential File Access

**WeDPHeaderCsReadS**

**Description**
Collectively reads the header information from a header file.

**Syntax**

```
WeDPHeaderCsReadS(ByVal FileName As String, ByVal SeriesNo As Long, ByRef ComBuff() As CommonInf1, ByRef ChBuff() As ChanellInf1)
```

**Return value**
Returns 0 if successful. Returns an error code if unsuccessful.

**Parameters**

- `FileName` *(IN)*: Header file name.
- `SeriesNo` *(IN)*: Sequence number when creating sequence files. If you specify -1, `FileName` becomes the file name as-is.
- `ComBuff()` *(OUT)*: Common information structure. Define using an array.
- `ChBuff()` *(OUT)*: Channel information structure. Define using an array.

**Note:**
Collectively retrieves the header file information for data of which the number of X-axis data points is the same. Prepare a single-element array for the common information structure buffer. Prepare channel information structure buffer for the amount equal to the number of channels.

**Example (Visual Basic)**

```
Dim ret As Long
Dim FileName As String
Dim SeriesNo As Long
Dim ComBuff(0) As CommonInf1
Dim ChBuff(5) As ChanellInf1
FileName = “TestData1”
BlockNo = 2
ret = WeDPHeaderCsReadS(FileName, SeriesNo, ComBuff(), ChBuff())
If 0 <> ret Then
    MsgBox “File read error”
End If
```

**Note**
This function collectively retrieves a section of the data in the header file as indicated in section 3.5, “Common Information Structure and Channel Information Structure.” To read unsupported items, use `WeDPHeaderItemRead` (page 3-19).

**WeDPCsRead**

**Description**
Reads the data from the data files (sequential files) by specifying the number of samples.

**Syntax**

```
WeDPCsRead(ByVal FileName As String, ByVal SeriesNo As Long, ByVal Start As Long, ByVal Length
As Long, ByVal ChNo As Long, ByVal DataForm As Long, ByRef DataBuff As Any)
```
Return value
Returns 0 if successful. Returns an error code if unsuccessful.

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FileName</td>
<td>Data file name. Specify the file name excluding the extension.</td>
</tr>
<tr>
<td>SeriesNo</td>
<td>Sequence number when reading sequential files. If you specify -1, only the file with the name specified by FileName is read.</td>
</tr>
<tr>
<td>Start</td>
<td>Start sample number. Specify the first sample number to be retrieved using a value greater than or equal to 0.</td>
</tr>
<tr>
<td>Length</td>
<td>Number of samples to be retrieved. -1 specifies all data after the sample specified by Start. An error occurs if the specified value is greater than the number of samples that is stored.</td>
</tr>
<tr>
<td>ChNo</td>
<td>Channel number. -1 specifies all channels. Counted from 1.</td>
</tr>
<tr>
<td>DataForm</td>
<td>DataBuff type. Specify BYTE (char), Integer (short), Long (long), Single (float), or Double (double). No other data types can be passed. The type inside the parentheses indicates the type for the C Language.</td>
</tr>
<tr>
<td>DataBuff</td>
<td>Buffer for storing data. Pass the buffer in the type specified by DataForm.</td>
</tr>
</tbody>
</table>

Note:
Reads the data from the data file by specifying the number of samples. Retrieves data of Length from sample Start in the file specified by FileName and SeriesNo. Specify DataForm from the following defined data.

<table>
<thead>
<tr>
<th>Type</th>
<th>DataForm</th>
</tr>
</thead>
<tbody>
<tr>
<td>BYTE</td>
<td>WE_UBYTE</td>
</tr>
<tr>
<td>Integer</td>
<td>WE_SWORD</td>
</tr>
<tr>
<td>Long</td>
<td>WE_SLONG</td>
</tr>
<tr>
<td>Single</td>
<td>WE_FLOAT</td>
</tr>
<tr>
<td>Double</td>
<td>WE_DOUBLE</td>
</tr>
</tbody>
</table>

Example (Visual Basic)

```vbnet
Dim FileName As String
Dim SeriesNo As Long
Dim Start As Long
Dim Length As Long
Dim ChNo As Long
Dim DataForm As Long
Dim DataBuff(1193,3) As Single
FileName = "TestData1"
ret = WeDPCsRead(FileName,2,100,1200,-1,WE_FLOAT,DataBuff(0,0))
If 0 <> ret Then
    MsgBox "File read error"
End If
```

WeDPHeaderCsWriteS

Description
Collectively writes the header information to the header file.

Syntax

```
WeDPHeaderCsWriteS(ByVal FileName As String, ByVal SeriesNo As Long, ByRef ComBuff() As CommInf1, ByRef ChBuff() As ChanelInf1, ByRef AcqInfo As AcqDataInfoEx)
```

Return value
Returns 0 if successful. Returns an error code if unsuccessful.
Parameters

FileName (IN)        Header file name.
SeriesNo (IN)        Sequence number when creating sequence files. If you specify -1, FileName becomes the file name as-is.
CommBuff() (IN)      Common information structure. Define using an array.
ChBuff() (IN)        Channel information structure. Define using an array.
AcqInfo (IN)         Data information structure.

Note:
Collectively writes the header file information for data of which the number of X-axis data points is the same. Prepare a single-element array for the common information structure buffer. Prepare channel information structure buffer for the amount equal to the number of channels. AcqInfo passes the data retrieved using WeAcqInfoInitialize(). Or, create the data by referring to section 3.5, “Common Information Structure and Channel Information Structure.” If the file already exists, it is overwritten.

Example (Visual Basic)
Dim ret As Long
Dim FileName As String
Dim SeriesNo As Long
Dim CommonBuff(0) As CommonInf1
Dim ChBuff(5) As ChanelInf1
Dim AcqInfo(5) As AcqDataInfoEx
Dim InfoNum As Long
Ret = WeGetAcqDataInfoEx(hMo,-1,5,AcqInfo(),infoNum)
If 0 <> ret Then
    MsgBox “Data information read error”
End If
FileName = “TestData1”
SeriesNo = 2
ret = WeDPHeaderWriteS(FileName,SeriesNo,CommonBuff(),ChBuff(), AcqInfo())
If 0 <> ret Then
    MsgBox “File write error”
End If

WeDPCsWrite
Description
Write data to a sequence file.

Syntax
WeDPCsWrite(ByVal FileName As String, ByVal SeriesNo As Long, ByVal SampleNum As Long, ByRef AcqInfo As Acq DataInfoEx, ByVal DataForm As Long, ByRef DataBuff() As Any)

Return value
Returns 0 if successful. Returns an error code if unsuccessful.

Parameters
FileName (IN)        Data file name. Specify the file name excluding the extension. A .wvf extension is added to the created file.
SeriesNo (IN)        Sequence number when creating sequence files. If you specify -1, FileName becomes the file name as-is.
SampleNum (IN)       Number of samples.
### AcqInfo (IN)
Data information structure. Or, create the data by referring to section 3.5, “Common Information Structure and Channel Information Structure.”

### DataForm (IN)
DataBuff type. Specify BYTE (char), Integer (short), Long (long), Single (float), or Double (double). No other data types can be passed. The type inside the parentheses indicates the type for the C Language.

### DataBuff (OUT)
Data storage buffer. Pass the buffer in the type specified by DataForm. Pass a buffer defined using a two-dimensional array.

#### Note:
Write data to a data file.

The files are handled as sequential files. SeriesNo is automatically added to the file names.

Specify DataForm from the following defined data.

<table>
<thead>
<tr>
<th>Type</th>
<th>Defined Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>BYTE</td>
<td>WE_UBYTE</td>
</tr>
<tr>
<td>Integer</td>
<td>WE_SWORD</td>
</tr>
<tr>
<td>Long</td>
<td>WE_SLONG</td>
</tr>
<tr>
<td>Single</td>
<td>WE_FLOAT</td>
</tr>
<tr>
<td>Double</td>
<td>WE_DOUBLE</td>
</tr>
</tbody>
</table>

#### Example (Visual Basic)
```vbnet
Dim FileName As String
Dim SeriesNo As Long
Dim AcqInfo(2) As AcqDataInfoEx
Dim DataForm As Long
Dim DataBuff(2,1000) As Single
Dim InfoNum As Integer
WeGetAcqDataInfoEx(hMo,-1,2,AcqInfo(),InfoNum)
FileName = “TestData1”
SeriesNo = 2
DataForm = WE_FLOAT
ret = WeDPCsWrite(FileName,SeriesNo,AcqInfo(),DataForm,
DataBuff(0,0))
If 0 <> ret Then
    MsgBox “File write error”
End If
```
3.9 Access the Specified Item of the Header File

WeDPHeaderItemRead

**Description**
Reads the information of the specified item name and specified channel from the header information of the header file.

**Syntax**

```vbnet
WeDPHeaderItemRead(ByVal FileName As String, ByVal ItemName As String, ByVal ChNo As Long, ByVal BlockNo As Long, ByRef DataBuff As String)
```

**Return value**
Returns 0 if successful. Returns an error code if unsuccessful.

**Parameters**
- `FileName` (IN) Data file name. Specify the file name excluding the extension.
- `ItemName` (IN) Item name.
- `ChNo` (IN) Channel number. Ignored if an item unrelated to the channel number is specified.
- `BlockNo` (IN) Block number.
- `DataBuff` (OUT) Buffer for storing data.

**Note:**
Retrieves the information of the specified item and specified channel from the header file information. You must have an understanding of the structure of the .hdr file when using this function. A block number is added to the item number for data containing multiple blocks. To read such item, specify the item name with the block number. Items “GroupName” and “TraceNumber” cannot be retrieved.

**Example (Visual Basic)**

```vbnet
Dim FileName As String
Dim ItemName As String
Dim ChNo As Long
Dim DataBuff As String*256
Dim BlockNo As Long
FileName = "TestData"
ItemName = "VResolution"
ChNo = 1
BlockNo = 0
Ret = WeDPHeaderItemRead(FileName, ItemName, ChNo, BlockNo, DataBuff)
If 0 <> ret Then
    MsgBox "Error in reading a header file item"
End If
```

WeDPHeaderItemWrite

**Description**
Writes data to the specified item name and specified channel in the header information of the header file.

**Syntax**

```vbnet
WeDPHeaderItemWrite(ByVal FileName As String, ByVal ItemName As String, ByVal ChNo As Long, ByVal BlockNo As Long, ByRef DataBuff As String)
```
Return value
Returns 0 if successful. Returns an error code if unsuccessful.

Parameters
FileName (IN) Data file name. Specify the file name excluding the extension.
ItemName (IN) Item name.
ChNo (IN) Channel number.
BlockNo (IN) Block number.
DataBuff (OUT) Buffer for storing data.

Note:
Writes data to the specified item and specified channel in the header information file.
Items “FormatVersion,” “Model,” “Endian,” “DataFormat,” “GroupNumber,” “TraceTotalNumber,”
“TraceName,” “BlockNumber,” and “VDataType” cannot be specified.
When you set PLinearMode, set to “0” as OFF or “1” as ON.

Example (Visual Basic)
Dim FileName As String
Dim ItemName As String
Dim ChNo As Long
Dim BlockNo As Long
Dim DataBuff As String
FileName = “TestData”
ItemName = “VResolution”
ChNo = 1
BlockNo = 0
DataBuff = CStr(5.42)
Ret = WeDPHeaderItemWrite(FileName,ItemName,ChNo,BlockNo,DataBuff)
If 0 <> ret Then
    MsgBox “Error in reading a header file item”
End If
3.10 Data Operation

WeDPGetSampleChNum

Description
Gets the number of samples and number of channels of the specified file.

Syntax
WeDPGetSampleChNum(ByVal FileName As String, ByVal BlockNo As Long, ByRef SampleNum As Long, ByRef ChNum As Long)

Return value
Returns 0 if successful. Returns an error code if unsuccessful.

Parameters
FileName (IN) Data file name. Specify the file name excluding the extension.
BlockNo (IN) Block number. -1 specifies all blocks.
SampleNum (OUT) Number of samples.
ChNum (OUT) Number of channels.

Note:
The number of samples and number of channels of the specified file are returned. For Scan type files, the total number of samples is returned regardless of the BlockNo setting.

Example (Visual Basic)
Dim FileName As String
Dim BlockNo As Long
Dim SampleNum As Long
Dim ChNum As Long
FileName = “TestData”
BlockNo = 0
ret = WeDPGetSampleChNum(FileName, SampleNum, ChNum)
If 0 <> ret Then
    MsgBox “Error in reading the number of samples and number of channels”
End If

WeDPGetBlockNum

Description
Gets the number of blocks of the specified file.

Syntax
WeDPGetBlockNum(ByVal FileName As String, ByRef BlockNum As Long)

Return value
Returns 0 if successful. Returns an error code if unsuccessful.

Parameters
FileName (IN) Data file name. Specify the file name excluding the extension.
BlockNum (OUT) Number of blocks.
**Note:**
The number of blocks of the specified file is returned.

**Example (Visual Basic)**
```
Dim FileName As String
Dim BlockNum As Long
FileName = "TestData"
ret = WeDPGetBlockNum(FileName, BlockNum)
If 0 <> ret Then
    MsgBox "Error in reading the number of blocks"
End If
```

**WeDPInitializeAcqInfo**

**Description**
Stores the required data in the data information structure.

**Syntax**
```
WeDPInitializeAcqInfo(ByVal VMaxData As Double, ByVal VMinData As Double, ByVal SampleNum As Long, ByVal SampInterval As Double, ByRef AcqInfo() As AcqDataInfoEx)
```

**Return value**
Returns 0 if successful. Returns an error code if unsuccessful.

**Parameters**
- **VMaxData (IN)** Scale Max.
- **VMinData (IN)** Scale Min.
- **SampleNum (IN)** Number of samples.
- **SampInterval (IN)** Sampling interval (s).
- **AcqInfo() (OUT)** Data information structure.

**Note:**
Sets and returns the required data in the data information structure.
The data set in AcqDataInfo by this function is as follows:
- **Chanel** 1 through n is set in the order of the array.
- **dataType** Data buffer type is set using the parameter of WeDPDataWrite or WeDPCsWrite.
- **blockNum** 1
- **startBit** 0
- **effectiveBit** 0
- **trigActive** 0
- **record** SampleNum.
- **recordLen** SampleNum.
- **time** 0
- **trigPosition** 0.0
- **Interval** Set to the SamplingInterval parameter.
- **VResolution** Set to 1.0.
- **VOffset** Set to 0.0.
- **TrigLevel** Set to 0.0.
- **TrigWidth** Set to 0.0.
- **PlusOverData** Set to the VMaxData parameter.
- **MinusOverData** Set to the VMinData parameter.
- **NonData** Set to the lost data.
- **DispMaxData** Set to the VMaxData parameter.
- **DispMinData** Set to the VMinData parameter.
Example (Visual Basic)

Dim VMaxData As Double
Dim VMinData As Double
Dim AcqInfo(3) As AcqDataInfoEx
VMaxData = 2.0
VMinData = -2.0
ret = WeDPIInitializeAcqInfo(VMaxData,VMinData,1000,0.001,AcqInfo())
If 0 <> ret Then
    MsgBox “Error in setting data to data information structure”
End If
### 3.11 The List of C Language Interface

#### Single File Access

<table>
<thead>
<tr>
<th>Function Name</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>WeDPHeaderReadS</td>
<td>int WeDPHeaderReadS(char* lpName, UINT blockno, LPSAFEARRAY* ComInfo, LPSAFEARRAY* ChBuff)</td>
</tr>
<tr>
<td>WeDPDataRead</td>
<td>int WeDPDataRead(char* lpName, int chno, UINT blockno, UINT dataForm, LPVOID dataBuff)</td>
</tr>
<tr>
<td>WeDPHeaderWriteS</td>
<td>int WeDPHeaderWriteS(char* lpName, UINT blockno, LPSAFEARRAY* ComBuff, LPSAFEARRAY* ChBuff, LPSAFEARRAY* AcqInfo)</td>
</tr>
<tr>
<td>WeDPDataWrite</td>
<td>int WeDPDataWrite(char* lpName, UINT chNum, UINT blockno, UINT SampleNum, UINT dataForm, LPVOID buf, LPSAFEARRAY* AcqData, LPVOID DataBuff)</td>
</tr>
</tbody>
</table>

#### Sequential File Access

<table>
<thead>
<tr>
<th>Function Name</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>WeDPHeaderCsReadS</td>
<td>int WeDPHeaderCsReadS(char* lpName, int seriesNo, LPSAFEARRAY* ComBuff, LPSAFEARRAY* ChBuff)</td>
</tr>
<tr>
<td>WeDPCsRead</td>
<td>int WeDPCsRead(char* lpName, int seriesNo, UINT start, int length, int chno, UINT dataForm, LPVOID dataBuff)</td>
</tr>
<tr>
<td>WeDPHeaderCsWriteS</td>
<td>int WeDPHeaderCsWriteS(char* lpName, int seriesNo, LPSAFEARRAY* ComBuff, LPSAFEARRAY* ChBuff, LPSAFEARRAY* AcqInfo)</td>
</tr>
<tr>
<td>WeDPCsWrite</td>
<td>int WeDPCsWrite(char* lpName, int seriesNo, LPSAFEARRAY* AcqData, UINT dataForm, LPVOID dataBuff)</td>
</tr>
</tbody>
</table>

#### Access the Specified Item of the Header File

<table>
<thead>
<tr>
<th>Function Name</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>WeDPHeaderItemRead</td>
<td>int WeDPHeaderItemRead(char* lpName, char* ItemName, UINT ChNo, UINT blockno, char* data)</td>
</tr>
<tr>
<td>WeDPHeaderItemWrite</td>
<td>int WeDPHeaderItemWrite(char* lpName, char* ItemName, UINT ChNo, UINT blockno, char* data)</td>
</tr>
</tbody>
</table>

#### Data Operation

<table>
<thead>
<tr>
<th>Function Name</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>WeDPGetSampleChNum</td>
<td>int WeDPGetSampleChNum(char* lpName, UINT blockno, UINT* SampleNum, UINT* ChNum)</td>
</tr>
<tr>
<td>WeDPGetBlockNum</td>
<td>int WeDPGetBlockNum(char* lpName, UINT blockNo)</td>
</tr>
<tr>
<td>WeDPIntializeAcqInfo</td>
<td>int WeDPIntializeAcqInfo(double VMaxData, double VMinData, double SamplingInterval, LPSAFEARRAY* AcqInfo)</td>
</tr>
</tbody>
</table>
## 4. Error Cord

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Error Code (Hex)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AsciiRSLT_NORMAL</td>
<td>0x0000</td>
<td>Normal completion.</td>
</tr>
<tr>
<td>AsciiRSLT_CANT_OPEN</td>
<td>0x0001</td>
<td>File open failure.</td>
</tr>
<tr>
<td>AsciiRSLT_CANT_ALLOC</td>
<td>0x0002</td>
<td>Memory allocation failure.</td>
</tr>
<tr>
<td>AsciiRSLT_FORMAT_ERROR</td>
<td>0x0003</td>
<td>The number of fields of the record is not correct.</td>
</tr>
<tr>
<td>AsciiRSLT_HANDLE_NULL</td>
<td>0x0004</td>
<td>NULL handle.</td>
</tr>
<tr>
<td>AsciiRSLT_NOT_INTEGER</td>
<td>0x0005</td>
<td>The data type is not UINT. GroupNumber, TraceTotalNumber, DataOffset, TraceNumber, BlockNumber, and BlockSize are UINT.</td>
</tr>
<tr>
<td>AsciiRSLT_POSITION_ERROR</td>
<td>0x0006</td>
<td>Description position is not correct. For example, a Group item is in PublicInfo.</td>
</tr>
<tr>
<td>AsciiRSLT_SCRIPT_ERROR</td>
<td>0x0007</td>
<td>Error in the item descriptor. For example, BlockSizeA.</td>
</tr>
<tr>
<td>AsciiRSLT_STRING_ERROR</td>
<td>0x0008</td>
<td>Error in Endian or DataFormat. Endian: Little or Big DataFormat: Trace or Block</td>
</tr>
<tr>
<td>AsciiRSLT_CANT_READ</td>
<td>0x0009</td>
<td>Failed to read. For example, &quot;?&quot;. Vresolution, Voffset, VUnit, VPlusOverData, VminusOverData, VilegalData, VMaxData, VMinData, HRresolution, HUnit, Date, or Time</td>
</tr>
<tr>
<td>AsciiRSLT_BOUNDARY</td>
<td>0x000A</td>
<td>Range designation error.</td>
</tr>
<tr>
<td>AsciiRSLT_READ_ERROR</td>
<td>0x000B</td>
<td>Access error to VWF file.</td>
</tr>
<tr>
<td>AsciiRSLT_UNSUPPORTED_DATATYPE</td>
<td>0x000C</td>
<td>Error in DataType. Only IU1, IS1, IU2, IS2, IU4, IS4, FS4, and FS8 are supported.</td>
</tr>
<tr>
<td>AsciiRSLT_LINK_ERROR</td>
<td>0x000D</td>
<td>DLL link failure.</td>
</tr>
<tr>
<td>AsciiRSLT_UNSUPPORTED_FUNCTION</td>
<td>0x000E</td>
<td>Function not supported by the specified DLL.</td>
</tr>
<tr>
<td>AsciiRSLT_ZERO_BLOCKSIZE</td>
<td>0x000F</td>
<td>The block size is 0.</td>
</tr>
<tr>
<td>AsciiRSLT_UNSUPPORTED_SCALING</td>
<td>0x0010</td>
<td>Scaling designation error. Other than AsciiADWave, AsciiPhysicalWave, or AsciiScalingWave.</td>
</tr>
<tr>
<td>AsciiRSLT_DISK_FULL</td>
<td>0x0011</td>
<td>Insufficient free disk space.</td>
</tr>
<tr>
<td>AsciiRSLT_MODIFY_ERROR</td>
<td>0x0012</td>
<td>Error in the parameter.</td>
</tr>
<tr>
<td>AsciiRSLT_ERROR</td>
<td>0x00FF</td>
<td>Other error.</td>
</tr>
</tbody>
</table>
Index

T
Terms and Conditions of the Software License ..................... 2

W
WeDPCsRead ................................................................. 3-15
WeDPCsWrite ............................................................. 3-17
WeDPDataRead ........................................................... 3-11
WeDPDataWrite .......................................................... 3-13
WeDPGetBlockNum ...................................................... 3-21
WeDPGetSampleChNum ............................................... 3-21
WeDPHeaderCsReadS .................................................... 3-15
WeDPHeaderCsWriteS .................................................... 3-16
WeDPHeaderItemRead ................................................... 3-19
WeDPHeaderItemWrite ................................................... 3-19
WeDPHeaderReadS ....................................................... 3-11
WeDPHeaderWriteS ....................................................... 3-12
WeDPIInitializeAcqInfo ............................................... 3-22