Akiyama Manufacturing

Serial Port Monitor and Analyzer Series

Protocol Analyzer for Modbus ASCII

Model: AKM-RSM-FM0

User's Manual

Version 1.3.0.0

TABLE OF CONTENTS

Precaution

[Trademark]

■ Modbus is a registered trademark of Schneider Electric, Inc.

■ CANopen is a registered trademark of CAN in Automation.

■ Windows is a registered trademark of Microsoft Corporation in the United States and other countries.

■ All other brand or product names are or may be trademarks or registered trademarks of, and are used to identify products or services of, their respective owners.

Caution:

(1) You must not reprint all (or a part) of the contents of this manual without getting the permission of Akiyama Manufacturing.

(2) The contents of this manual may be changed in the future without a notice.

Reference document:

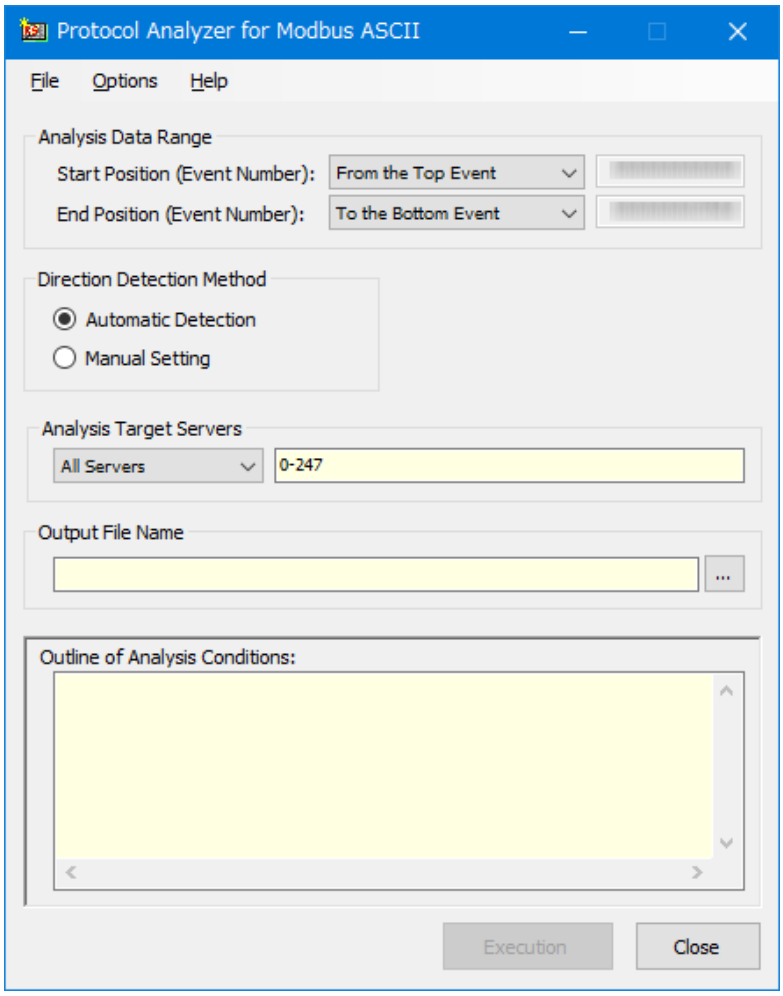
MODBUS APPLICATION PROTOCOL SPECIFICATION V1.1b3

Introduction

The Protocol Analyzer for Modbus ASCII (Model: AKM-RSM-FM0) is the tool which analyzes the Monitor Result data of the Serial Port Monitor and Analyzer (Model: AKM-RSM-100). The analysis result is outputted to the text file.
From now on, " Protocol Analyzer for Modbus ASCII (Model: AKM-RSM-FM0)" will be described as "Protocol Analyzer".
And also, "Serial Port Monitor and Analyzer (Model: AKM-RSM-100)" will be described as "AKM-RSM-100".

Analyzed contents are based on V1.1b3 of Modbus.

The flow of the communications, the wrong point about the protocol, and so on will be confirmed visually by analyzing Monitor Result data.



Screen image

The Protocol Analyzer is the Add-on function of the AKM-RSM-100.
AKM-RSM-100 Professional Edition is necessary to use this Protocol Analyzer.
(Not available in Basic Edition and Standard Edition of AKM-RSM-100.)

The Protocol Analyzer has two analysis types of the "Detailed Analysis" and the "Outline Analysis".

■ Detailed Analysis

All data string is analyzed in detail. In addition, relations between the [Request] from the Client and the [Response] from the Server are confirmed.
Then, Alert is outputted when some problems are found.

```
                                        ⋮

---- [Normal] [Request] (0x000000000001) (xxxx/xx/xx xx:xx:xx) ---------------------------------->
:010100000001FD<cr><lf>

Server Address: 1 (0x01), Function Number: 1 (0x01)
Function Name : Read Coils

Starting Address: 0x0001, Quantity of Coils: 1 (0x0001)
------------------------------------------------------------------------------------------------->

<--- [Normal] [Response] (0x000000000013) (xxxx/xx/xx xx:xx:xx) ----------------------------------
:01010105F8<cr><lf>

Server Address: 1 (0x01), Function Number: 1 (0x01)
Function Name : Read Coils

Byte Count: 1 (0x01)
Status    : ----- Following --------------------
0x0001:ON
                                                        (Request Event Number: 0x000000000001)
<-------------------------------------------------------------------------------------------------

---- [Normal] [Request] (0x000000000021) (xxxx/xx/xx xx:xx:xx) ---------------------------------->
:010100130013D8<cr><lf>

Server Address: 1 (0x01), Function Number: 1 (0x01)
Function Name : Read Coils

Starting Address: 0x0014, Quantity of Coils: 19 (0x0013)
------------------------------------------------------------------------------------------------->

<--- [Normal] [Response] (0x000000000032) (xxxx/xx/xx xx:xx:xx) ----------------------------------
:010103CD6B05BE<cr><lf>

Server Address: 1 (0x01), Function Number: 1 (0x01)
Function Name : Read Coils

Byte Count: 3 (0x03)
Status    : ----- Following --------------------
0x0014:ON , 0x0015:OFF, 0x0016:ON , 0x0017:ON , 0x0018:OFF, 0x0019:OFF, 0x001A:ON , 0x001B:ON ,
0x001C:ON , 0x001D:ON , 0x001E:OFF, 0x001F:ON , 0x0020:OFF, 0x0021:ON , 0x0022:ON , 0x0023:OFF,
0x0024:ON , 0x0025:OFF, 0x0026:ON
                                                        (Request Event Number: 0x000000000021)
<-------------------------------------------------------------------------------------------------

                                        ⋮
```

File output image (Detailed Analysis)

■ Outline Analysis

Only the Server Address and the Function Number are analyzed.

```
                                        ⋮

---- [Normal] [Request] (0x000000000001) (xxxx/xx/xx xx:xx:xx) ---------------------------------->
:010100000001FD<cr><lf>
Server Address: 1 (0x01), Function Number: 1 (0x01)
Function Name : Read Coils
------------------------------------------------------------------------------------------------>

<--- [Normal] [Response] (0x000000000013) (xxxx/xx/xx xx:xx:xx) ----------------------------------
:01010105F8<cr><lf>
Server Address: 1 (0x01), Function Number: 1 (0x01)
Function Name : Read Coils
<------------------------------------------------------------------------------------------------

---- [Normal] [Request] (0x000000000021) (xxxx/xx/xx xx:xx:xx) ---------------------------------->
:010100130013D8<cr><lf>
Server Address: 1 (0x01), Function Number: 1 (0x01)
Function Name : Read Coils
------------------------------------------------------------------------------------------------>

<--- [Normal] [Response] (0x000000000032) (xxxx/xx/xx xx:xx:xx) ----------------------------------
:010103CD6B05BE<cr><lf>
Server Address: 1 (0x01), Function Number: 1 (0x01)
Function Name : Read Coils
<------------------------------------------------------------------------------------------------

---- [Normal] [Request] (0x000000000044) (xxxx/xx/xx xx:xx:xx) ---------------------------------->
:0101000007D027<cr><lf>
Server Address: 1 (0x01), Function Number: 1 (0x01)
Function Name : Read Coils
------------------------------------------------------------------------------------------------>

                                        ⋮
```

File output image (Outline Analysis)

Precautions

(1) The Protocol Analyzer **can be started from [Tools] menu of the AKM-RSM-100 Professional Edition**. (Not available in Basic Edition and Standard Edition of AKM-RSM-100.)

(2) The Protocol Analyzer does the analysis based on the specification of the **"Modbus Application Protocol Specification V1.1b3"**.
A proper analysis result may not be able to get it in the case of the data string of the other version of Modbus.

(3) When the [Request] from the Client and the [Response] from the Server is being mixed in the Monitor Result data, **a proper analysis result may not be able to get it**.
(Especially, when [Request] and [Response] are only in either the Port 1 or the Port 2 in the RS-485 network of 2 wires through both.)

(4) When there is the [Stop mark] in the Analysis Data Range, **the Analysis is stopped with the [Stop mark]**.

(5) The Protocol Analyzer is application software based on **.NetFramework4**.
If you are using the AKM-RSM-100 of the Version 4 series, confirm that .NetFramework4 is in your PC.
If .NetFramework4 is not in your PC, install it in advance before the Protocol Analyzer is started.

(6) **An Analysis Result File may become enormous volume** corresponding to the volume of the Monitor Result data.
Therefore, output an Analysis Result File to the storage of NTFS.

Operational Conditions

(1) Start Protocol Analyzer in accordance with "How to Start" which is described later from [Tools] menu of AKM-RSM-100.

(2) The Protocol Analyzers can start multiple from AKM-RSM-100.
However, be careful of the number of the starting because the memory resource of PC (Windows) is occupied corresponding to the number of the Protocol Analyzer which was started.

(3) When the Protocol Analyzer was started, following contents are taken over from the setting of the AKM-RSM-100.

- Top Event Number of Monitor Result data
- Current Cursor Position of Monitor Result data
- Bottom Event Number of Monitor Result data
- Date Format
- Display Condition of Tool Tip

(4) When the AKM-RSM-100 Professional Edition is being used by the **User ID of the Trial edition**, **only 10 Protocol data string will be analyzed**. **Then, processing will be stopped**.

(5) In this manual, each procedure and each screen display are being described based on Windows 10.
In other OS, each procedure and each screen display are almost same as Windows 10.

How to Start

The Protocol Analyzer is the Add-on Program of the AKM-RSM-100.
Therefore, the Protocol Analyzer is started from the [Tools] menu of the AKM-RSM-100.
First, the Protocol Analyzer must be registered in advance in the [Tools] menu of the AKM-RSM-100.

When Protocol Analyzer is installed, "Add-on Menu Control" will be actuated. And, it will merge Protocol Analyzer to the [Tools] menu of the AKM-RSM-100. However, in the case of following, "Add-on Menu Control" will be not able to merge menu.

- When AKM-RSM-100 isn't being installed.
- When 10 kinds of Add-on is registered to the menu already.

*Note: Data will be replaced when Add-on of same name was registered on the [Tools] menu.

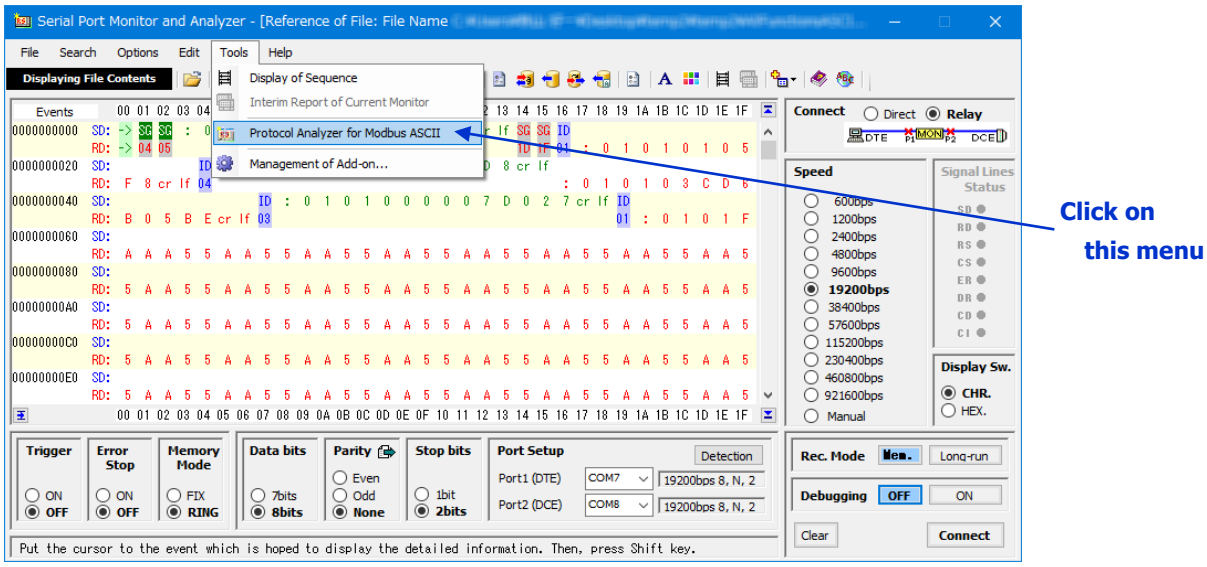
In these cases, the menu of the Protocol Analyzer must be registered by using [Tools]-[Management of Add-on] on the AKM-RSM-100 by the manual operation. Contents of registration by the manual operation are the followings.

- Menu Name : Protocol Analyzer for Modbus ASCII
- Execution File Name : [Install Folder] \ AKM-RSM-FM0.exe
- CMD-line Argument : /MULTI <%datafile%>.aod
- Data File Name : AKM-RSM-FM0_<%date%>

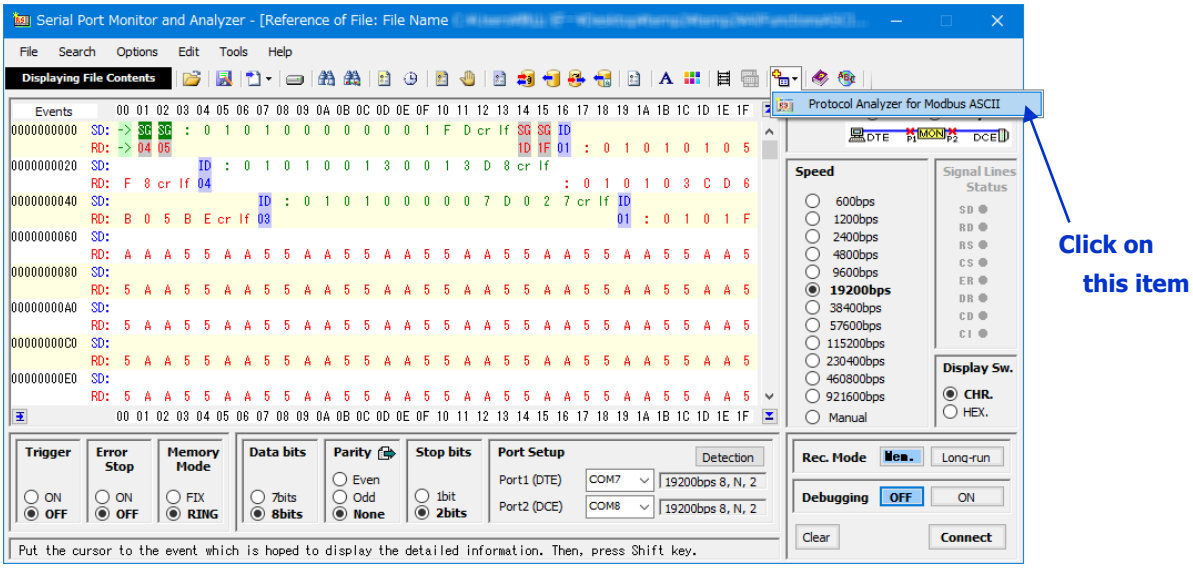
*Note: Usually, the default setting of Install Folder is the following.
[OS Drive]: \ Program Files (x86) \ Serial Port Monitor and Analyzer \ Add-on \AKM-RSM-FM0

From now on, various descriptions will be described under the assumption that the registration of the Add-on menu is completed. Start Protocol Analyzer as follows from AKM-RSM-100.

How to start: Click on [Tools]-[Protocol Analyzer for Modbus ASCII] of Menu bar.



Or drop down  icon of Tool bar, and click on [Protocol Analyzer for Modbus ASCII].



The Protocol Analyzer is available when the status of the AKM-RSM-100 is the following.

Displaying Monitor Result	Displaying File Contents
Displaying Monitor Result	Displaying File Contents

How to Operate

■ Open the Monitor Result File

When AKM-RSM-100 is the following status, the Monitor Result File must be opened by the manual operation first.

No Data and Disconnected　No Data and Connected

By clicking on [File]-[Open] of Menu bar, open the Monitor Result File to be displayed.
After that, start the Protocol Analyzer.

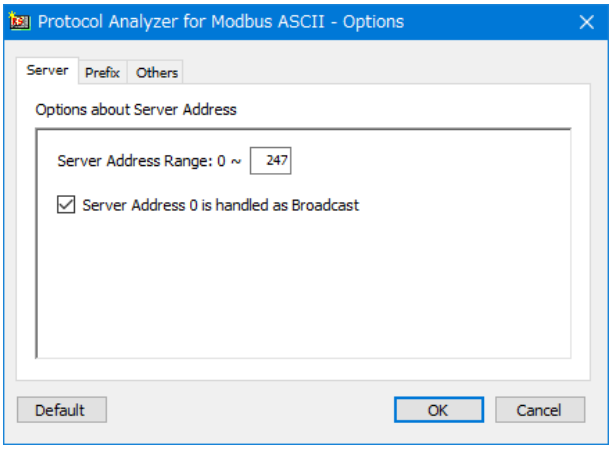
■ Specify Option Setting

First, specify various Option items of the Protocol Analyzer.
Click on [Options]-[Options] of Menu bar of the Protocol Analyzer. In this operation, [Options] screen will be displayed.

Close Options screen by clicking on OK button after all designation is completed.

(1) Server Option

Specify about the Server Address here.



[Server Address Range]: (Default: 0-247)

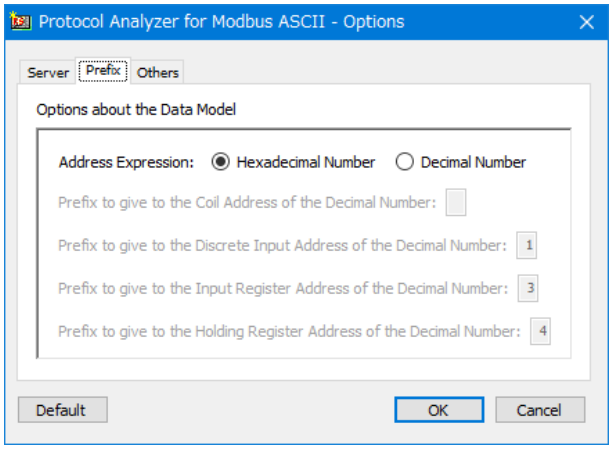
Specify the range of Server Address which is being used in your system.
Maximum number of server is 255.

[Server Address 0 is handled as Broadcast]: (Default: Checked)

Check it when the Server Address 0 is the Broadcast.
The Function [Request] of the Broadcast is processed as the communications of No Response.
If the Server Address 0 is handled in the same way as other Server, remove this check.

(2) Prefix Option

Specify about the Data Model here.



[Address Expression]: (Default: Hexadecimal Number)

Specify the expression of each following Address which is used in the Analysis Result File.

(a) Coil Address
(b) Discrete Input Address
(c) Input Register Address
(d) Holding Register Address

[Prefix to give to the Coil Address of the Decimal Number]: (Default: "" No Prefix)

Specify Prefix which is added to the head of Coil Address when the Coil Address is outputted by the Decimal Number.

No Prefix or the Decimal Number "0" to "9" is available.

[Prefix to give to the Discrete Input Address of the Decimal Number]: (Default: "1")

Specify Prefix which is added to the head of Discrete Input Address when the Discrete Input Address is outputted by the Decimal Number.

No Prefix or the Decimal Number "0" to "9" is available.

[Prefix to give to the Input Register Address of the Decimal Number]: (Default: "3")

Specify Prefix which is added to the head of Input Register Address when the Input Register Address is outputted by the Decimal Number.

No Prefix or the Decimal Number "0" to "9" is available.

[Prefix to give to the Holding Register Address of the Decimal Number]: (Default: "4")

Specify Prefix which is added to the head of Holding Register Address when the Holding Register Address is outputted by the Decimal Number.

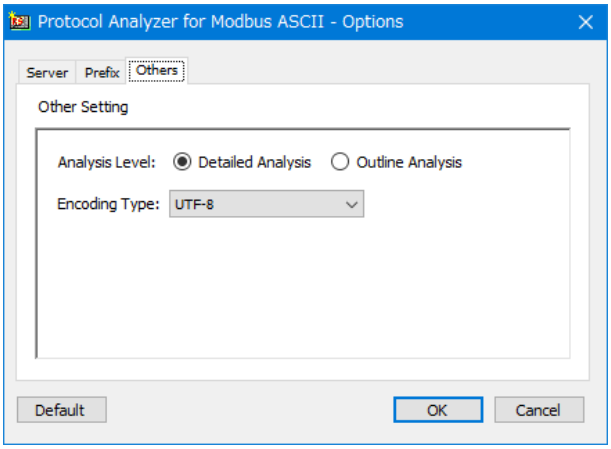
No Prefix or the Decimal Number "0" to "9" is available.

*Note: Each Address exists from 1 to 65536 in the decimal number. When Prefix is given, it is expressed from P0001 to P65536 (P: Prefix).

Example) When the Prefix is "3": from 30001 to 365536

(3) Others Option

Specify about the Other Setting here.



[Analysis Level]: (Default: Detailed Analysis)

Specify the Analysis Level.

[Detailed Analysis] : All data will be analyzed in detail.

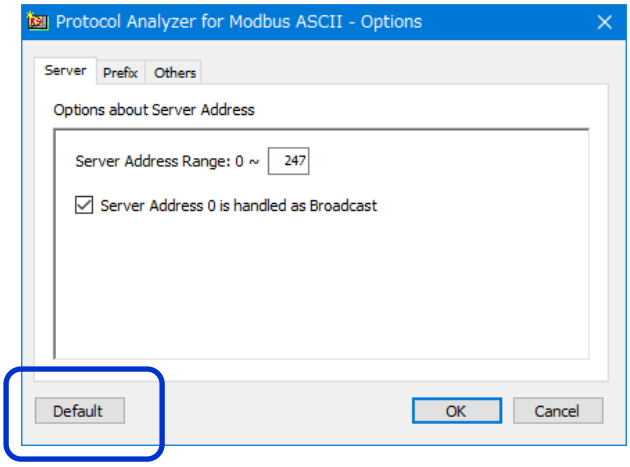
[Outline Analysis] : Only the Server Address and the Function Number (and Sub-Function Number and MEI Type) will be analyzed.

[Encoding Type]: (Default: UTF-8)

Specify encoding type of the Analysis Result File.

(4) [Default] Button

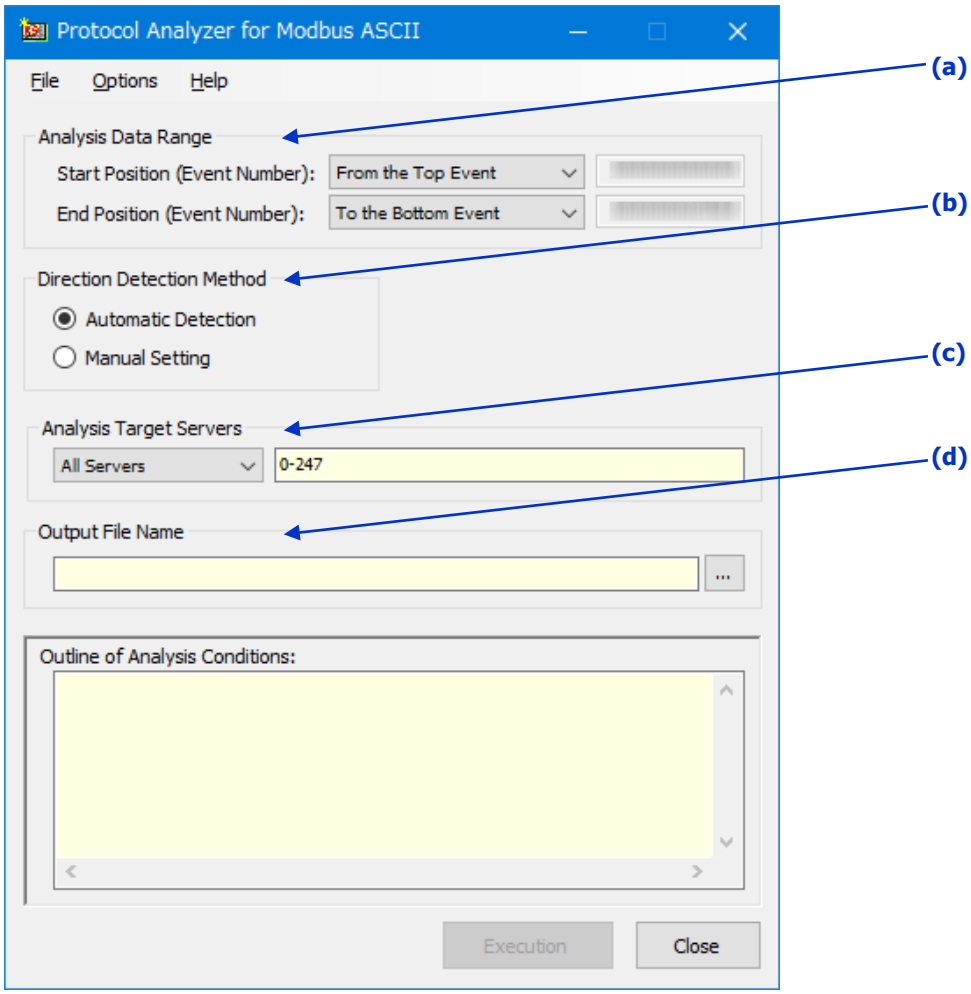
By clicking on this button, all Option setting is returned to the Default Setting.



*Note: After this button was clicked, the changes are applied by clicking on the [OK].

■ Execute the Protocol Analyzer

After setting up various Options, set up the screen of the Protocol Analyzer. After that, execute the Protocol Analyzer.



(a) [Analysis Data Range]:

[Start Position]: (Default: From the Top Event)

Specify the Start Event Number of Monitor Result data which is analyzed.

Specify it from three kinds of the following.

- [From the Top Event]
- [From the Current Cursor]
- [Manual Setting] (Input Event Number by manual operation.)

[End Position]: (Default: To the Bottom Event)

Specify the End Event Number of Monitor Result data which is analyzed.

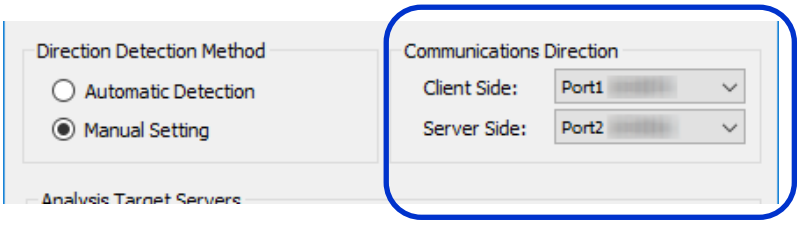
Specify it from three kinds of the following.

- [To the Bottom Event]
- [To the Current Cursor]
- [Manual Setting] (Input Event Number by manual operation.)

(b) [Direction Detection Method]: (Default: Automatic Detection)

Specify the detection method about the communications direction.
The Automatic Detection will find the Client Side port and the Server Side port automatically at the top process of the analysis execution by using the delivered Monitor Result Data.

Specify Client Side port and Server Side port by using manual operation when Client Side port and Server Side port can't find it by the Automatic Detection.
By specifying the Manual Setting, Communications Direction setting screen of the following figure will appear. Specify the Communications Direction here by the manual operation.

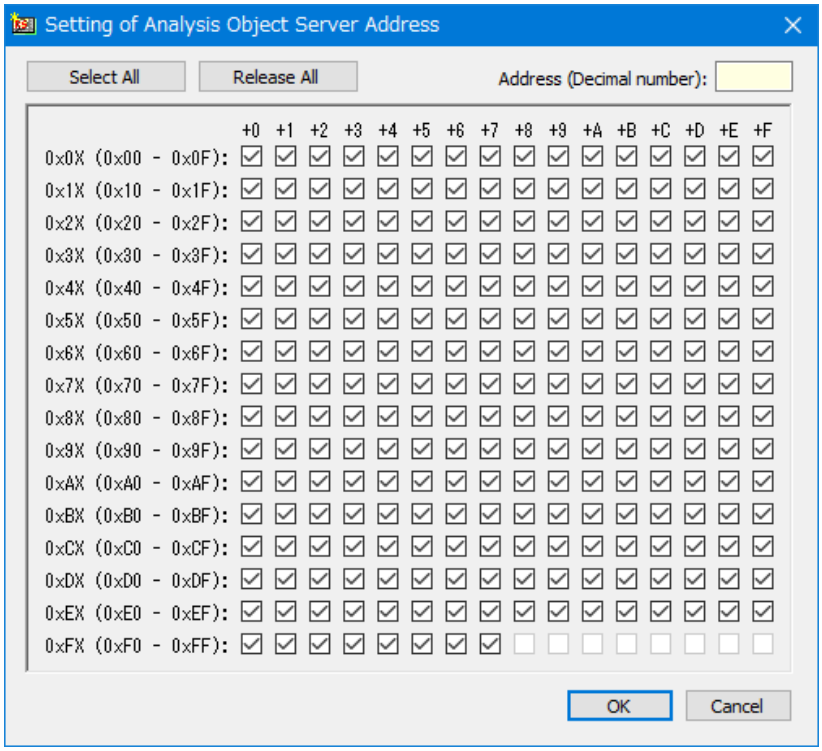


In addition, the Client Side port is port which transmits Function (request) of Modbus
And, the Server Side port is the port which transmits response by Function (request) from the Client Side.

(c) [Analysis Target Servers]: (Default: All Servers 0-247)

Specify the server addresses to analyze.

- [All Servers] : [Server Address Range] which was specified in the [Option] setting.
- [Choose Server] : The Server Address of the analysis object can do selection in the [Server Address Range] which was specified in the Option setting.
  When [Choose Server] is specified, the following [Setting of Analysis Object Server Address] screen is displayed.
  Give a check to the Server of the analysis object in this screen.
  Then, click on [OK] button.



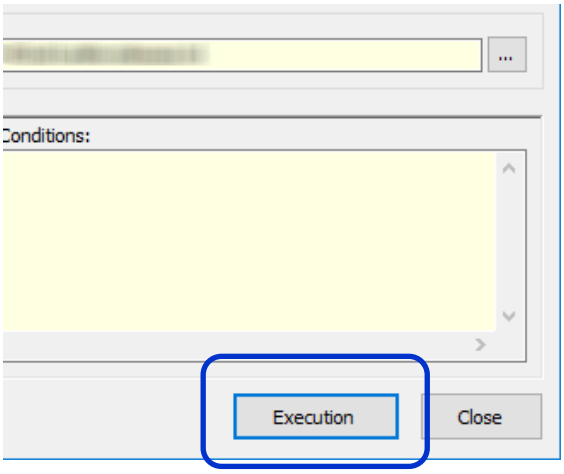
(d) [Output File Name]: (Default: "" (Null))

Specify the Path and File Name of Analysis Result File.

Click on [Text Box] or [...] button. Then, specify Drive, Folder, and File Name.

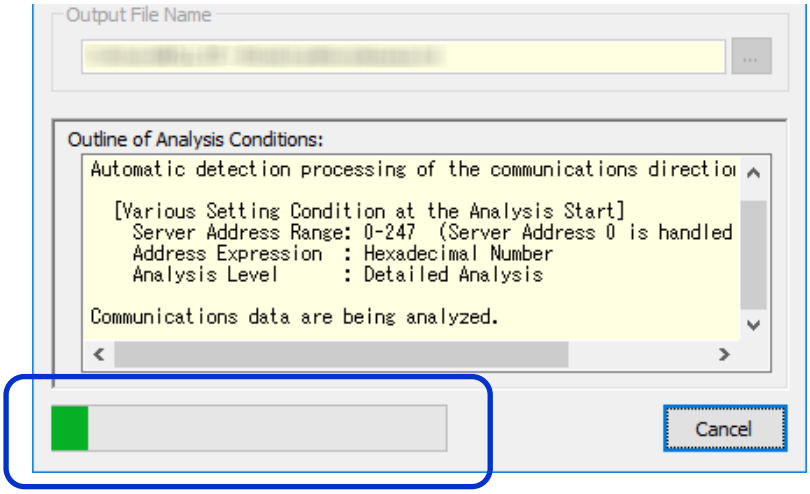
*Note: The direct input to the [Text Box] can't be done.

After [Output File Name] is specified, [Execute] button will be changed to active.



By clicking on Execute button, the analysis will be started.

Progress is displayed during analysis by the following Progress Bar. Wait until the analysis is completed.



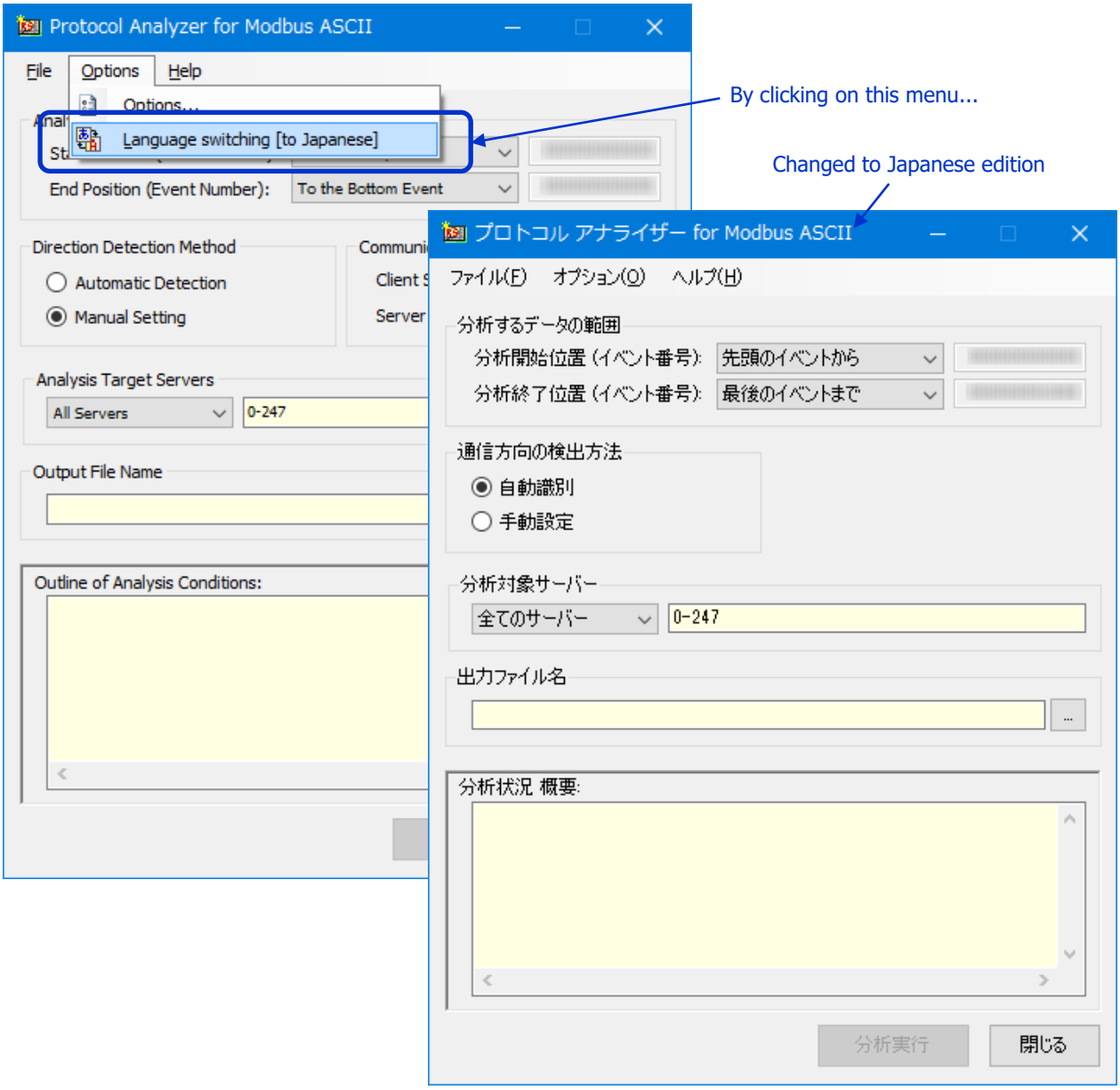
The analysis can be aborted during analysis processing by clicking on Cancel button.

Other Function

*Caution: When OS is Windows XP, character corruption will occur in this function.
This function shouldn't be used with Windows XP.

The Protocol Analyzer copes with both of Japanese and English. At first, the Protocol Analyzer is started in the language which is the same as the language of the AKM-RSM.
Then, the Analysis Result File is made in the language which is the same as the display language.

However, the display language of the Protocol Analyzer can be changed in the [Option] menu.
This function is useful when an analysis result file is provided to the Japanese technician.

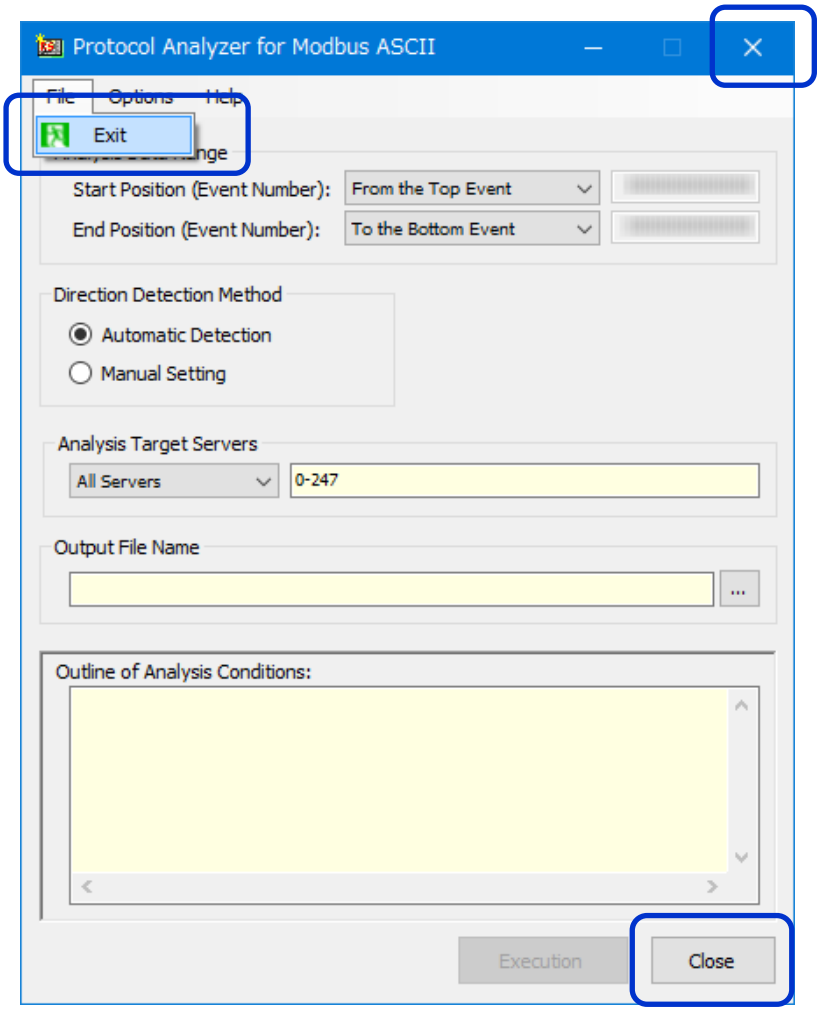


*Note: After it is changed to Japanese edition, it is returned to English edition by the same operation.

Shutdown method

Shut the Protocol Analyzer down in either next method.

Method 1: Click on [Close] of the lower right part
Method 2: Click on [File]-[Exit] of Menu bar
Method 3: Click on [×] of Title bar



About the contents of the Analysis Result File

The Analysis Result File is made by **Text File.**

The system image in the file is the following.

There is the Client in the Left hand. And, there is the Server in the Right hand.

The following is the example of the Analysis Result File of the Function 07.

```
Serial Port Monitor and Analyzer (Model: AKM-RSM-100)
Protocol Analyzer for Modbus ASCII (Ver.x.x.x.x)  [based on 1.1b3]  (Dic. Rev. xxxx/xx/xx)

Source File Name: xxxxxxxxxxxx.mon

  [Various Setting Condition at the Analysis Start]
    Server Address Range: 0-247  (Server Address 0 is handled as Broadcast: Yes)
    Address Expression  : Hexadecimal Number
    Analysis Level      : Detailed Analysis

    Analysis Target Server(s): 0-247

    Analysis Data Range      : 0x000000000000-0x000000000019


==== Analysis was started (xxxx/xx/xx xx:xx:xx) ==================================================

---- [Normal] [Request] (0x000000000003) (xxxx/xx/xx xx:xx:xx) ---------------------------------->
:1007E9<cr><lf>

Server Address: 16 (0x10), Function Number: 7 (0x07)
Function Name : Read Exception Status
------------------------------------------------------------------------------------------------->

<--- [Normal] [Response] (0x00000000000F) (xxxx/xx/xx xx:xx:xx) ----------------------------------
:10076D7C<cr><lf>

Server Address: 16 (0x10), Function Number: 7 (0x07)
Function Name : Read Exception Status

Output Data: OFF ON  ON  OFF ON  ON  OFF ON   (109 (0x6D))
                                                        (Request Event Number: 0x000000000003)
<-------------------------------------------------------------------------------------------------

Processing was completed normally.

==== Analysis was completed (xxxx/xx/xx xx:xx:xx) ================================================
```

(a)

(b)

The **(a)** is Header. This is the information which is useful when an Analysis Result File is referred to later.

The **(b)** is the Analysis Result. (This part may become enormous volume according to the volume of the Monitor Result data.)

■ About the Header

The configuration of the Header is the following.

```
Serial Port Monitor and Analyzer (Model: AKM-RSM-100)
Protocol Analyzer for Modbus ASCII (Ver.x.x.x.x)  [based on 1.1b3]  (Dic. Rev. xxxx/xx/xx)

Source File Name: xxxxxxxxxxxx.mon

  [Various Setting Condition at the Analysis Start]
    Server Address Range: 0-247  (Server Address 0 is handled as Broadcast: Yes)
    Address Expression  : Hexadecimal Number
    Analysis Level      : Detailed Analysis

    Analysis Target Server(s): 0-247

    Analysis Data Range      : 0x000000000000-0x000000000019
```

(1) : Parent Application Name of the Protocol Analyzer

(2)-1 : Name and Version of the Protocol Analyzer
(2)-2 : The Version of the Standard Protocol which the Protocol Analyzer is based on
(2)-3 : Revision date of Dictionary File for the Protocol Analyzer

(3) : The Monitor Result File which is analysis object (When the analysis object was Memory, it is outputted with [Memory].)

(4) : [Option] setting conditions when the Protocol Analyzer was executed

(5) : [Analysis Target Servers] when the Protocol Analyzer was executed

(6) : [Analysis Data Range] when the Protocol Analyzer was executed

This information is useful when an analysis result is referred to later. The conditions when an analysis was executed are definite by referring to this information.

■ About the Analysis Result

(1) Outline of the Analysis Result File Format

```
==== Analysis was started (xxxx/xx/xx xx:xx:xx) ==============================================

---- [Normal] [Request] (0x000000000003) (xxxx/xx/xx xx:xx:xx) ---------------------------->
:1007E9<cr><lf>  <-- One Protocol Data String (Character String)

Server Address: 16 (0x10), Function Number: 7 (0x07)
Function Name : Read Exception Status
---------------------------------------------------------------------------------------------->

<--- [Normal] [Response] (0x00000000000F) (xxxx/xx/xx xx:xx:xx) ----------------------------
:10076D7C<cr><lf>  <-- One Protocol Data String (Character String)

Server Address: 16 (0x10), Function Number: 7 (0x07)
Function Name : Read Exception Status

Output Data: OFF ON  ON  OFF ON  ON  OFF ON   (109 (0x6D))
                                                (Request Event Number: 0x000000000003)
<---------------------------------------------------------------------------------------------

Processing was completed normally.  <-- Analysis Processing Result

==== Analysis was completed (xxxx/xx/xx xx:xx:xx) ==============================================
```

Protocol Information

Request

Protocol Information

Response

Analysis Result

The [Request] from the Client is surrounded by the arrow (---------->).
And, the [Response] from the Server is surrounded by the arrow (<----------).

Following information is outputted in the first line of the [Request] and the [Response].

```
---- [Analysis Result] [Request]  (Head Event Number) (Time Stamp) ----------->
<--- [Analysis Result] [Response]  (Head Event Number) (Time Stamp) -----------
```

Analysis Result:

Normal	Data String is Normal. And there is no contradiction in the contents of Data String also.
Error!	Data String is Normal. However, there is contradiction in the contents of the Data String.
	The Data String which couldn't be analyzed.
Alert	[Request] was transmitted from Client to Server though there was a [Request] that [Response] hadn't been returned yet. *Note: Alert isn't outputted in the case shown below. ・Function which [Response] doesn't exist in. (Ex. Function 8 (Sub. 4)) ・When a Server Address 0 is handled as a Broadcast.
	Server transmitted [Response] though there was no [Request] from Client.
	[Response] of Server is different from the Client's [Request] Function.
Exception	Data String is Normal. And, normal Exception as the Protocol was returned. *Note: Exception occurs only in the [Response].

Head Event Number:

It is the Head Event Number (Hexadecimal Number) of this Protocol Data String.

Time Stamp:

It is the Time Stamp of this Protocol Data String.

*Note: When a Time Stamp isn't included in the Monitor Result data, "????/??/?? ??:??:??" will be outputted.

(2) Example of the Analysis Result of [Error!]

The Analysis Result is outputted as follows when the Data String is Normal and there is contradiction in the contents of the Data String.

```
---- [Error!] [Request] (0x000000000275) (xxxx/xx/xx xx:xx:xx) ---------------------------------->
:0101FFFF0002FE<cr><lf>

Server Address: 1 (0x01), Function Number: 1 (0x01)
Function Name : Read Coils

Starting Address: 0x10000, Quantity of Coils: 2 (0x0002)

*** The Starting Address or the Quantity of Coils is wrong. ***   <- Error Message
------------------------------------------------------------------------------------------------->
```

This is the example that the Client is trying to read two Coils from the Coil Address 65536.

The Data String which can't be analyzed is outputted as follows.

```
---- [Error!] [Request] (0x000000000207) (xxxx/xx/xx xx:xx:xx) ---------------------------------->
:010800130000E4<cr><lf>
*** Data string which couldn't be analyzed (Cause: xxxxxxxxxx) ***   <- Error Message
------------------------------------------------------------------------------------------------->
```

The Function 8 (Sub-Function 19) is reserved function.

The following will be outputted as the Cause.

- Un-available character is included in the ADU
- Number of byte of PDU is the Odd Number
- Illegal Header or Footer
- Un-supported Function
- Abnormal condition about the Data Length
- Unforeseen Error

In addition, the Data String is outputted simply as follows when the length of the Data String which can't be analyzed is longer than the length of the maximum ADU length (513 bytes).

```
---- [Error!] [Obscurity] (0x0000000000FC) (xxxx/xx/xx xx:xx:xx) ---------------------------------
3A-30-31-30-31-30-30-30-30-30-30-30-30-31-46-44-3A-30-31-30-31-30-30-30-30-30-30-30-30-31-46-44-3A-30-
        :
        : (827 Bytes)                                          } Simple output of the Data String
        :
-3A-30-31-30-31-30-30-30-30-30-30-30-30-31-46-44-3A-30-31-30-31-30-30-30-30-30-30-30-30-31-46-44-0D-0A
*** Data string which couldn't be analyzed (Cause: xxxxxxxxxx) ***   <- Error Message
------------------------------------------------------------------------------------------------------
```

*Note: About various Error Messages, refer to the Appendix A: Message List for Analysis Result File.

(3) Example of the Analysis Result of [Alert]

The Analysis Result is outputted as follows when [Request] was transmitted from Client to Server though there was a [Request] that [Response] hadn't been returned yet.

```
---- [Normal] [Request] (0x000000000001) (xxxx/xx/xx xx:xx:xx) ---------------------------------->
:010100000001FD<cr><lf>

Server Address: 1 (0x01), Function Number: 1 (0x01)
Function Name : Read Coils

Starting Address: 0x0001, Quantity of Coils: 1 (0x0001)
------------------------------------------------------------------------------------------------->

<--- [Alert] [Response] ------------------------------------------------------------------------
*** The Server 1 (0x01) hasn't returned response about the Last Request from the Client. ***
        Last Request: Read Coils
        Event Number: 0x000000000001  Request Data: xxxx.xx.xx xx:xx:xx
---- [Normal] [Request] (0x000000000014) (xxxx/xx/xx xx:xx:xx) ---------------------------------->
:010100130013D8<cr><lf>

Server Address: 1 (0x01), Function Number: 1 (0x01)
Function Name : Read Coils

Starting Address: 0x0014, Quantity of Coils: 19 (0x0013)
------------------------------------------------------------------------------------------------->
```

Alert Message

Alert Object

The Analysis Result is outputted as follows when Server transmitted [Response] though there was no [Request] from Client.

```
---- [Alert] [Request] ------------------------------------------------------------------------>
*** The Client hasn't transmitted the Request about the next Response. ***
<--- [Normal] [Response] (0x00000000001E) (xxxx/xx/xx xx:xx:xx) ---------------------------------
:010103CD6B05BE<cr><lf>

Server Address: 1 (0x01), Function Number: 1 (0x01)
Function Name : Read Coils

Byte Count: 3 (0x03)
Status    : ----- Following --------------------
????+0:ON , ????+1:OFF, ????+2:ON , ????+3:ON , ????+4:OFF, ????+5:OFF, ????+6:ON , ????+7:ON ,
????+8:ON , ????+9:ON , ???+10:OFF, ???+11:ON , ???+12:OFF, ???+13:ON , ???+14:ON , ???+15:OFF,
???+16:ON , ???+17:OFF, ???+18:ON , ???+19:OFF, ???+20:OFF, ???+21:OFF, ???+22:OFF, ???+23:OFF
<-------------------------------------------------------------------------------------------------
```

Alert Message

Alert Object

In this example, because there is no [Request], the Coil Address is unknown.

The Analysis Result is outputted as follows when [Response] of Server is different from the Client's [Request] Function.

```
---- [Normal] [Request] (0x000000000001) (xxxx/xx/xx xx:xx:xx) ---------------------------------->
:010100000001FD<cr><lf>

Server Address: 1 (0x01), Function Number: 1 (0x01)
Function Name : Read Coils

Starting Address: 0x0001, Quantity of Coils: 1 (0x0001)
------------------------------------------------------------------------------------------------->

---- [Alert] [Request] ------------------------------------------------------------------------->
*** The Client requested Function 1 in the Event Number 0x000000000001. ***
<--- [Normal] [Response] (0x000000000014) (xxxx/xx/xx xx:xx:xx) ----------------------------------
01020105F7<cr><lf>

Server Address: 1 (0x01), Function Number: 2 (0x02)
Function Name : Read Discrete Inputs

Byte Count: 1 (0x01)
Status    : ----- Following --------------------
????+0:ON , ????+1:OFF, ????+2:ON , ????+3:OFF, ????+4:OFF, ????+5:OFF, ????+6:OFF, ????+7:OFF
<------------------------------------------------------------------------------------------------
```

Alert Message

Alert Object

In this example, Server sent the response of Function 2 as the response of Function 1 which Client required.

(4) Example of the Analysis Result of [Exception]

```
<--- [Exception] [Response] (0x000000000286) (xxxx/xx/xx xx:xx:xx) ------------------------------
:0181017D<cr><lf>

Server Address: 1 (0x01), Function Number: 1 + 128 (0x01 + 0x80)
Function Name : Read Coils

Exception Code: 1 (0x01)  ILLEGAL FUNCTION
                                                    (Request Event Number: 0x000000000275)
<------------------------------------------------------------------------------------------------
```

[Exception] will be outputted as follows in each Exception Code.

Exception Code	Output Message
01	ILLEGAL FUNCTION
02	ILLEGAL DATA ADDRESS
03	ILLEGAL DATA VALUE
04	SERVER DEVICE FAILURE
05	ACKNOWLEDGE
06	SERVER DEVICE BUSY
08	MEMORY PARITY ERROR
0A	GATEWAY PATH UNAVAILABLE
0B	GATEWAY TARGET DEVICE FAILED TO RESPOND

*Note: In the case of the Exception Code except for the above table, it will be outputted as the "(UNKNOWN EXCEPTION)".

Appendix A: Message List for Analysis Result File

■ Error Message

Function	Direction	Error Message
01	Request	The Starting Address or the Quantity of Coils is wrong.
		The number of the read of Coil is over range.
	Response	There is contradiction in the number of Coil between Request and Response.
02	Request	The Starting Address or the Quantity of Coils is wrong.
		The number of the read of Discrete Input is over range.
	Response	There is contradiction in the number of Discrete Input between Request and Response.
03	Request	The Starting Address or the Quantity of Coils is wrong.
		The number of the read of Holding Register is over range.
	Response	Illegal Byte Count. (Odd number)
		There is contradiction in the number of Holding Register between Request and Response.
04	Request	The Starting Address or the Quantity of Coils is wrong.
		The number of the read of Input Register is over range.
	Response	Illegal Byte Count. (Odd number)
		There is contradiction in the number of Input Register between Request and Response.
05	Request	Illegal Output Value.
	Response	The written Address is different from the Address of Request.
		Illegal Output Value.
06	Response	The written Address is different from the Address of Request.
08-00	Response	Response Data are different from the Request Data.
08-01	Response	Illegal Process Type.
08-02	Request	Illegal Value of Data.
08-03	Request	Illegal Value of Data (Second Byte).
	Response	Response Delimiter data are different from the Request Delimiter data.
		Illegal Value of Data (Second Byte).
08-04	Request	Illegal Value of Data.
08-10	Request	Illegal Value of Data.
	Request	Illegal Value of Data.
08-11	Request	Illegal Value of Data.
08-12	Request	Illegal Value of Data.
08-13	Request	Illegal Value of Data.
08-14	Request	Illegal Value of Data.
08-15	Request	Illegal Value of Data.
08-16	Request	Illegal Value of Data.
08-17	Request	Illegal Value of Data.
08-18	Request	Illegal Value of Data.
08-20	Request	Illegal Value of Data.
	Response	Illegal Value of Data.

Function	Direction	Error Message
12	Response	Illegal Data Length of the Protocol Data.
		Illegal Byte Count.
15	Request	There is contradiction between Quantity of Coils and Byte Count.
		The Starting Address or the Quantity of Coils is wrong.
		The number of the write of Coil is over range.
	Response	The written Address is different from the Address of Request.
		The written Quantity of Coils is different from the Quantity of Coils of Request.
16	Request	The Starting Address or the Quantity of Registers is wrong.
		Illegal Byte Count. (Odd number)
		There is contradiction between Quantity of Registers and Byte Count.
	Response	The written Address is different from the Address of Request.
		The written Quantity of Registers is different from the Quantity of Registers of Request.
20	Request	Illegal Byte Count.
		Illegal Byte Count.
		Illegal File Number (0). (Group x)
		The number of the read of Record is over range. (Group x)
		The Starting Record Number is over range. (Group x)
		Illegal Reference Type. (Group x)
	Response	There will be the Group of abnormal Byte Count. (Lack of the communications data)
		There is contradiction in the number of Record between Request and Response. (Group x)
		Illegal Byte Count. (Even number) (Group x)
		Illegal Reference Type. (Group x)
21	Request	There will be the Group of abnormal Byte Count. (Lack of the communications data)
		Illegal Byte Count. (Even number)
		The number of the write of Record is over range. (Group x)
		The Starting Record Number is over range. (Group x)
		Illegal File Number (0). (Group x)
		Illegal Reference Type. (Group x)
	Response	There will be the Group of abnormal Byte Count. (Lack of the communications data)
		The written Record Data is different from the Record Data of Request. (Group x)
		The written Address is different from the Address of Request. (Group x)
		The written Starting Record Number is different from the Request. (Group x)
		The written File Number is different from the File Number of Request. (Group x)
		Illegal Byte Count. (Even number)
		The number of the write of Record is over range. (Group x)
		The Starting Record Number is over range. (Group x)
		Illegal File Number (0). (Group x)
		Illegal Reference Type. (Group x)
22	Response	T The OR Mask is different from the OR Mask of Request.
		The AND Mask is different from the AND Mask of Request.
		The Reference Address is different from the Reference Address of Request.

Function	Direction	Error Message
23	Request	Illegal Byte Count.
		The number of the write of Register is over range.
		The number of the read of Register is over range.
		The number of the write of Register is over range.
		The number of the read of Register is over range.
	Response	There is contradiction in the Quantity to Read between Request and Response.
		Illegal Byte Count. (Odd number)
24	Response	Illegal Data Length of the Protocol Data.
		Illegal Byte Count. (Un-match FIFO Count)
		Illegal Byte Count. (Odd number)
43-14	Request	Illegal Read Device ID Code.
	Response	Illegal Data Length of the Protocol Data.
		Illegal Next Object ID.
		Illegal More Follows data.
		Illegal Conformity Level.
		Illegal Read Device ID Code.

Common		The Data Length of this Protocol Data is longer than PDU of the Standard Specification.
		Illegal LRC value.

Exception	Response	In the Standard Specification, this Exception isn't supposed to occur in this Function.

■ Alert Message

Direction	Alert Message
Request	The Server x (0xXX) hasn't returned response about the Last Request from the Client.
Response	The Client hasn't transmitted the Request about the next Response.
	The Client requested Function x in the Event Number 0xXXXXXXXXXXXX.

Appendix B: Example of Analysis Result in each Function

Function 1

```
---- [Normal] [Request] (0x000000000021) (xxxx/xx/xx xx:xx:xx) ---------------------------------->
:010100130013D8<cr><lf>

Server Address: 1 (0x01), Function Number: 1 (0x01)
Function Name : Read Coils

Starting Address: 0x0014, Quantity of Coils: 19 (0x0013)
------------------------------------------------------------------------------------------------->

<--- [Normal] [Response] (0x000000000032) (xxxx/xx/xx xx:xx:xx) ----------------------------------
:010103CD6B05BE<cr><lf>

Server Address: 1 (0x01), Function Number: 1 (0x01)
Function Name : Read Coils

Byte Count: 3 (0x03)
Status    : ----- Following --------------------
0x0014:ON , 0x0015:OFF, 0x0016:ON , 0x0017:ON , 0x0018:OFF, 0x0019:OFF, 0x001A:ON , 0x001B:ON ,
0x001C:ON , 0x001D:ON , 0x001E:OFF, 0x001F:ON , 0x0020:OFF, 0x0021:ON , 0x0022:ON , 0x0023:OFF,
0x0024:ON , 0x0025:OFF, 0x0026:ON
                                                          (Request Event Number: 0x000000000021)
<-------------------------------------------------------------------------------------------------
```

Function 2

```
---- [Normal] [Request] (0x000000000025) (xxxx/xx/xx xx:xx:xx) ---------------------------------->
:010200C4001623<cr><lf>

Server Address: 1 (0x01), Function Number: 2 (0x02)
Function Name : Read Discrete Inputs

Starting Address: 0x00C5, Quantity of Coils: 22 (0x0016)
------------------------------------------------------------------------------------------------->

<--- [Normal] [Response] (0x000000000037) (xxxx/xx/xx xx:xx:xx) ----------------------------------
:010203ACDE353B<cr><lf>

Server Address: 1 (0x01), Function Number: 2 (0x02)
Function Name : Read Discrete Inputs

Byte Count: 3 (0x03)
Status    : ----- Following --------------------
0x00C5:OFF, 0x00C6:OFF, 0x00C7:ON , 0x00C8:ON , 0x00C9:OFF, 0x00CA:ON , 0x00CB:OFF, 0x00CC:ON ,
0x00CD:OFF, 0x00CE:ON , 0x00CF:ON , 0x00D0:ON , 0x00D1:ON , 0x00D2:OFF, 0x00D3:ON , 0x00D4:ON ,
0x00D5:ON , 0x00D6:OFF, 0x00D7:ON , 0x00D8:OFF, 0x00D9:ON , 0x00DA:ON
                                                          (Request Event Number: 0x000000000025)
<-------------------------------------------------------------------------------------------------
```

Function 3

```
---- [Normal] [Request] (0x000000000027) (xxxx/xx/xx xx:xx:xx) ---------------------------------->
:0103006B00038E<cr><lf>

Server Address: 1 (0x01), Function Number: 3 (0x03)
Function Name : Read Holding Registers

Starting Address: 0x006C, Quantity of Coils: 3 (0x0003)
------------------------------------------------------------------------------------------------->

<--- [Normal] [Response] (0x000000000039) (xxxx/xx/xx xx:xx:xx) ----------------------------------
:010306022B0000006465<cr><lf>

Server Address: 1 (0x01), Function Number: 3 (0x03)
Function Name : Read Holding Registers

Byte Count    : 6 (0x06)
Register Value: ----- Following --------------------
0x006C:   555 (0x022B), 0x006D:     0 (0x0000), 0x006E:   100 (0x0064)
                                                        (Request Event Number: 0x000000000027)
<-------------------------------------------------------------------------------------------------
```

Function 4

```
---- [Normal] [Request] (0x000000000001) (xxxx/xx/xx xx:xx:xx) ---------------------------------->
:0104FFFD0003FC<cr><lf>

Server Address: 1 (0x01), Function Number: 4 (0x04)
Function Name : Read Input Registers

Starting Address: 0xFFFE, Quantity of Coils: 3 (0x0003)
------------------------------------------------------------------------------------------------->

<--- [Normal] [Response] (0x000000000013) (xxxx/xx/xx xx:xx:xx) ----------------------------------
:010406123456789ABC8B<cr><lf>

Server Address: 1 (0x01), Function Number: 4 (0x04)
Function Name : Read Input Registers

Byte Count    : 6 (0x06)
Register Value: ----- Following --------------------
0xFFFE:  4660 (0x1234), 0xFFFF: 22136 (0x5678), 0x10000: 39612 (0x9ABC)
                                                        (Request Event Number: 0x000000000001)
<-------------------------------------------------------------------------------------------------
```

Function 5

```
---- [Normal] [Request] (0x000000000028) (xxxx/xx/xx xx:xx:xx) ---------------------------------->
:010500ACFF004F<cr><lf>

Server Address: 1 (0x01), Function Number: 5 (0x05)
Function Name : Write Single Coil

Address: 0x00AD, Output Value: ON (0xFF00)
------------------------------------------------------------------------------------------------>

<--- [Normal] [Response] (0x00000000003A) (xxxx/xx/xx xx:xx:xx) ---------------------------------
:010500ACFF004F<cr><lf>

Server Address: 1 (0x01), Function Number: 5 (0x05)
Function Name : Write Single Coil

Address: 0x00AD, Output Value: ON (0xFF00)
                                                                (Request Event Number: 0x000000000028)
<------------------------------------------------------------------------------------------------
```

Function 6

```
---- [Normal] [Request] (0x000000000025) (xxxx/xx/xx xx:xx:xx) ---------------------------------->
:010600010003F5<cr><lf>

Server Address: 1 (0x01), Function Number: 6 (0x06)
Function Name : Write Single Register

Address: 0x0002, Register Value: 3 (0x0003)
------------------------------------------------------------------------------------------------>

<--- [Normal] [Response] (0x000000000036) (xxxx/xx/xx xx:xx:xx) ---------------------------------
:010600010003F5<cr><lf>

Server Address: 1 (0x01), Function Number: 6 (0x06)
Function Name : Write Single Register

Address: 0x0002, Register Value: 3 (0x0003)
                                                                (Request Event Number: 0x000000000025)
<------------------------------------------------------------------------------------------------
```

Function 7

```
---- [Normal] [Request] (0x000000000003) (xxxx/xx/xx xx:xx:xx) ---------------------------------->
:1007E9<cr><lf>

Server Address: 16 (0x10), Function Number: 7 (0x07)
Function Name : Read Exception Status
------------------------------------------------------------------------------------------------>

<--- [Normal] [Response] (0x00000000000F) (xxxx/xx/xx xx:xx:xx) ---------------------------------
:10076D7C<cr><lf>

Server Address: 16 (0x10), Function Number: 7 (0x07)
Function Name : Read Exception Status

Output Data: OFF ON  ON  OFF ON  ON  OFF ON   (109 (0x6D))
                                                                (Request Event Number: 0x000000000003)
<------------------------------------------------------------------------------------------------
```

Function 8-0

```
---- [Normal] [Request] (0x000000000001) (xxxx/xx/xx xx:xx:xx) ---------------------------------->
:01080000414274<cr><lf>

Server Address: 1 (0x01), Function Number: 8-0 (0x08-0x0000)
Function Name : Diagnostics - Return Query Data

Data: 65 66 (0x41 0x42)
------------------------------------------------------------------------------------------------->

<--- [Normal] [Response] (0x000000000013) (xxxx/xx/xx xx:xx:xx) ----------------------------------
:01080000414274<cr><lf>

Server Address: 1 (0x01), Function Number: 8-0 (0x08-0x0000)
Function Name : Diagnostics - Return Query Data

Data: 65 66 (0x41 0x42)
                                                        (Request Event Number: 0x000000000001)
<-------------------------------------------------------------------------------------------------
```

Function 8-1

```
---- [Normal] [Request] (0x000000000025) (xxxx/xx/xx xx:xx:xx) ---------------------------------->
:010800010000F6<cr><lf>

Server Address: 1 (0x01), Function Number: 8-1 (0x08-0x0001)
Function Name : Diagnostics - Restart Communications Option

Process Type: 0 (0x0000)  Normal Restart of communications port.
------------------------------------------------------------------------------------------------->

<--- [Normal] [Response] (0x000000000037) (xxxx/xx/xx xx:xx:xx) ----------------------------------
:010800010000F6<cr><lf>

Server Address: 1 (0x01), Function Number: 8-1 (0x08-0x0001)
Function Name : Diagnostics - Restart Communications Option

Process Type: 0 (0x0000)  Normal Restart of communications port.
                                                        (Request Event Number: 0x000000000025)
<-------------------------------------------------------------------------------------------------
```

Function 8-2

```
---- [Normal] [Request] (0x00000000006C) (xxxx/xx/xx xx:xx:xx) ---------------------------------->
:010800020000F5<cr><lf>

Server Address: 1 (0x01), Function Number: 8-2 (0x08-0x0002)
Function Name : Diagnostics - Return Diagnostic Register

Data: 0 (0x0000)
------------------------------------------------------------------------------------------------->

<--- [Normal] [Response] (0x00000000007E) (xxxx/xx/xx xx:xx:xx) ----------------------------------
:010800021020C5<cr><lf>

Server Address: 1 (0x01), Function Number: 8-2 (0x08-0x0002)
Function Name : Diagnostics - Return Diagnostic Register

Diagnostic Register Contents: 4128 (0x1020)
                                                        (Request Event Number: 0x00000000006C)
<-------------------------------------------------------------------------------------------------
```

Function 8-3

```
---- [Normal] [Request] (0x000000000090) (xxxx/xx/xx xx:xx:xx) ---------------------------------->
:010800030A00EA<cr><lf>

Server Address: 1 (0x01), Function Number: 8-3 (0x08-0x0003)
Function Name : Diagnostics - Change ASCII Input Delimiter

Message Delimiter: "lf" (10 (0x0A))
------------------------------------------------------------------------------------------------->

<--- [Normal] [Response] (0x0000000000A2) (xxxx/xx/xx xx:xx:xx) ----------------------------------
:010800030A00EA<cr><lf>

Server Address: 1 (0x01), Function Number: 8-3 (0x08-0x0003)
Function Name : Diagnostics - Change ASCII Input Delimiter

Message Delimiter: "lf" (10 (0x0A))
                                                                  (Request Event Number: 0x000000000090)
<-------------------------------------------------------------------------------------------------
```

Function 8-4

```
---- [Normal] [Request] (0x0000000000B4) (xxxx/xx/xx xx:xx:xx) ---------------------------------->
:010800040000F3<cr><lf>

Server Address: 1 (0x01), Function Number: 8-4 (0x08-0x0004)
Function Name : Diagnostics - Force Listen Only Mode

Data: 0 (0x0000)
------------------------------------------------------------------------------------------------->
```

*Note: There is no Response in the Function 8-4.

Function 8-10

```
---- [Normal] [Request] (0x0000000000C6) (xxxx/xx/xx xx:xx:xx) ---------------------------------->
:0108000A0000ED<cr><lf>

Server Address: 1 (0x01), Function Number: 8-10 (0x08-0x000A)
Function Name : Diagnostics - Clear Counters and Diagnostic Register

Data: 0 (0x0000)
------------------------------------------------------------------------------------------------->

<--- [Normal] [Response] (0x0000000000D7) (xxxx/xx/xx xx:xx:xx) ----------------------------------
:0108000A0000ED<cr><lf>

Server Address: 1 (0x01), Function Number: 8-10 (0x08-0x000A)
Function Name : Diagnostics - Clear Counters and Diagnostic Register

Data: 0 (0x0000)
                                                                  (Request Event Number: 0x0000000000C6)
<-------------------------------------------------------------------------------------------------
```

Function 8-11

```
---- [Normal] [Request] (0x0000000000E9) (xxxx/xx/xx xx:xx:xx) ---------------------------------->
:0108000B0000EC<cr><lf>

Server Address: 1 (0x01), Function Number: 8-11 (0x08-0x000B)
Function Name : Diagnostics - Return Bus Message Count

Data: 0 (0x0000)
------------------------------------------------------------------------------------------------->

<--- [Normal] [Response] (0x0000000000FB) (xxxx/xx/xx xx:xx:xx) ----------------------------------
:0108000B20309C<cr><lf>

Server Address: 1 (0x01), Function Number: 8-11 (0x08-0x000B)
Function Name : Diagnostics - Return Bus Message Count

Total Message Count: 8240 (0x2030)
                                                        (Request Event Number: 0x0000000000E9)
<-------------------------------------------------------------------------------------------------
```

Function 8-12

```
---- [Normal] [Request] (0x00000000010D) (xxxx/xx/xx xx:xx:xx) ---------------------------------->
:0108000C0000EB<cr><lf>

Server Address: 1 (0x01), Function Number: 8-12 (0x08-0x000C)
Function Name : Diagnostics - Return Bus Communication Error Count

Data: 0 (0x0000)
------------------------------------------------------------------------------------------------->

<--- [Normal] [Response] (0x00000000011F) (xxxx/xx/xx xx:xx:xx) ----------------------------------
:0108000C30407B<cr><lf>

Server Address: 1 (0x01), Function Number: 8-12 (0x08-0x000C)
Function Name : Diagnostics - Return Bus Communication Error Count

CRC Error Count: 12352 (0x3040)
                                                        (Request Event Number: 0x00000000010D)
<-------------------------------------------------------------------------------------------------
```

Function 8-13

```
---- [Normal] [Request] (0x000000000131) (xxxx/xx/xx xx:xx:xx) ---------------------------------->
:0108000D0000EA<cr><lf>

Server Address: 1 (0x01), Function Number: 8-13 (0x08-0x000D)
Function Name : Diagnostics - Return Bus Exception Error Count

Data: 0 (0x0000)
------------------------------------------------------------------------------------------------->

<--- [Normal] [Response] (0x000000000143) (xxxx/xx/xx xx:xx:xx) ----------------------------------
:0108000D40505A<cr><lf>

Server Address: 1 (0x01), Function Number: 8-13 (0x08-0x000D)
Function Name : Diagnostics - Return Bus Exception Error Count

Exception Error Count: 16464 (0x4050)
                                                        (Request Event Number: 0x000000000131)
<-------------------------------------------------------------------------------------------------
```

Function 8-14

```
---- [Normal] [Request] (0x000000000155) (xxxx/xx/xx xx:xx:xx) ---------------------------------->
:0108000E0000E9<cr><lf>

Server Address: 1 (0x01), Function Number: 8-14 (0x08-0x000E)
Function Name : Diagnostics - Return Server Message Count

Data: 0 (0x0000)
------------------------------------------------------------------------------------------------->

<--- [Normal] [Response] (0x000000000167) (xxxx/xx/xx xx:xx:xx) ----------------------------------
:0108000E506039<cr><lf>

Server Address: 1 (0x01), Function Number: 8-14 (0x08-0x000E)
Function Name : Diagnostics - Return Server Message Count

Server Message Count: 20576 (0x5060)
                                                                  (Request Event Number: 0x000000000155)
<-------------------------------------------------------------------------------------------------
```

Function 8-15

```
---- [Normal] [Request] (0x000000000179) (xxxx/xx/xx xx:xx:xx) ---------------------------------->
:0108000F0000E8<cr><lf>

Server Address: 1 (0x01), Function Number: 8-15 (0x08-0x000F)
Function Name : Diagnostics - Return Server No Response Count

Data: 0 (0x0000)
------------------------------------------------------------------------------------------------->

<--- [Normal] [Response] (0x00000000018A) (xxxx/xx/xx xx:xx:xx) ----------------------------------
:0108000F607018<cr><lf>

Server Address: 1 (0x01), Function Number: 8-15 (0x08-0x000F)
Function Name : Diagnostics - Return Server No Response Count

Server No Response Count: 24688 (0x6070)
                                                                  (Request Event Number: 0x000000000179)
<-------------------------------------------------------------------------------------------------
```

Function 8-16

```
---- [Normal] [Request] (0x00000000019C) (xxxx/xx/xx xx:xx:xx) ---------------------------------->
:010800100000E7<cr><lf>

Server Address: 1 (0x01), Function Number: 8-16 (0x08-0x0010)
Function Name : Diagnostics - Return Server NAK Count

Data: 0 (0x0000)
------------------------------------------------------------------------------------------------->

<--- [Normal] [Response] (0x0000000001AE) (xxxx/xx/xx xx:xx:xx) ----------------------------------
:010800107080F7<cr><lf>

Server Address: 1 (0x01), Function Number: 8-16 (0x08-0x0010)
Function Name : Diagnostics - Return Server NAK Count

Server NAK Count: 28800 (0x7080)
                                                                  (Request Event Number: 0x00000000019C)
<-------------------------------------------------------------------------------------------------
```

Function 8-17

```
---- [Normal] [Request] (0x0000000001C0) (xxxx/xx/xx xx:xx:xx) ---------------------------------->
:010800110000E6<cr><lf>

Server Address: 1 (0x01), Function Number: 8-17 (0x08-0x0011)
Function Name : Diagnostics - Return Server Busy Count

Data: 0 (0x0000)
------------------------------------------------------------------------------------------------->

<--- [Normal] [Response] (0x0000000001D1) (xxxx/xx/xx xx:xx:xx) ----------------------------------
:010800118090D6<cr><lf>

Server Address: 1 (0x01), Function Number: 8-17 (0x08-0x0011)
Function Name : Diagnostics - Return Server Busy Count

Server Device Busy Count: 32912 (0x8090)
                                                        (Request Event Number: 0x0000000001C0)
<-------------------------------------------------------------------------------------------------
```

Function 8-18

```
---- [Normal] [Request] (0x0000000001E3) (xxxx/xx/xx xx:xx:xx) ---------------------------------->
:010800120000E5<cr><lf>

Server Address: 1 (0x01), Function Number: 8-18 (0x08-0x0012)
Function Name : Diagnostics - Return Bus Character Overrun Count

Data: 0 (0x0000)
------------------------------------------------------------------------------------------------->

<--- [Normal] [Response] (0x0000000001F5) (xxxx/xx/xx xx:xx:xx) ----------------------------------
:0108001290A0B5<cr><lf>

Server Address: 1 (0x01), Function Number: 8-18 (0x08-0x0012)
Function Name : Diagnostics - Return Bus Character Overrun Count

Server Character Overrun Count: 37024 (0x90A0)
                                                        (Request Event Number: 0x0000000001E3)
<-------------------------------------------------------------------------------------------------
```

Function 8-20

```
---- [Normal] [Request] (0x00000000022B) (xxxx/xx/xx xx:xx:xx) ---------------------------------->
:010800140000E3<cr><lf>

Server Address: 1 (0x01), Function Number: 8-20 (0x08-0x0014)
Function Name : Diagnostics - Clear Overrun Counter and Flag

Data: 0 (0x0000)
------------------------------------------------------------------------------------------------->

<--- [Normal] [Response] (0x00000000023D) (xxxx/xx/xx xx:xx:xx) ----------------------------------
:010800140000E3<cr><lf>

Server Address: 1 (0x01), Function Number: 8-20 (0x08-0x0014)
Function Name : Diagnostics - Clear Overrun Counter and Flag

Data: 0 (0x0000)
                                                        (Request Event Number: 0x00000000022B)
<-------------------------------------------------------------------------------------------------
```

Function 11

```
---- [Normal] [Request] (0x000000000003) (xxxx/xx/xx xx:xx:xx) ---------------------------------->
:100BE5<cr><lf>

Server Address: 16 (0x10), Function Number: 11 (0x0B)
Function Name : Get Comm Event Counter
------------------------------------------------------------------------------------------------->

<--- [Normal] [Response] (0x00000000000E) (xxxx/xx/xx xx:xx:xx) ----------------------------------
:100B0000FFFFE7<cr><lf>

Server Address: 16 (0x10), Function Number: 11 (0x0B)
Function Name : Get Comm Event Counter

Status     : 0 (0x0000)  The remote device is not processing a program function.
Event Count: 65535 (0xFFFF)
                                                                (Request Event Number: 0x000000000003)
<-------------------------------------------------------------------------------------------------
```

Function 12

```
---- [Normal] [Request] (0x000000000023) (xxxx/xx/xx xx:xx:xx) ---------------------------------->
:110CE3<cr><lf>

Server Address: 17 (0x11), Function Number: 12 (0x0C)
Function Name : Get Comm Event Log
------------------------------------------------------------------------------------------------->

<--- [Normal] [Response] (0x00000000002D) (xxxx/xx/xx xx:xx:xx) ----------------------------------
:110C08000001080121200090<cr><lf>

Server Address: 17 (0x11), Function Number: 12 (0x0C)
Function Name : Get Comm Event Log

Byte Count : 8 (x008)
Status     : 0 (0x0000)  The remote device is not processing a program function.
Event Count: 264 (0x0108), Message Count: 289 (0x0121)
Event Log  : ----- Following (Event 0 is most recent communications event.) --------------------
Event   0: 32 (0x20)
Event   1: 0 (0x00)  Remote device Initiated Communication Restart
                                                                (Request Event Number: 0x000000000023)
<-------------------------------------------------------------------------------------------------
```

Function 15

```
---- [Normal] [Request] (0x00000000002C) (xxxx/xx/xx xx:xx:xx) ---------------------------------->
:010F0013000A02CD0103<cr><lf>

Server Address: 1 (0x01), Function Number: 15 (0x0F)
Function Name : Write Multiple Coils

Starting Address: 0x0014, Quantity of Coils: 10 (0x000A)
Byte Count      : 2 (0x02)
Status          : ----- Following --------------------
0x0014:ON , 0x0015:OFF, 0x0016:ON , 0x0017:ON , 0x0018:OFF, 0x0019:OFF, 0x001A:ON , 0x001B:ON ,
0x001C:ON , 0x001D:OFF
------------------------------------------------------------------------------------------------>

<--- [Normal] [Response] (0x000000000044) (xxxx/xx/xx xx:xx:xx) ----------------------------------
:010F0013000AD3<cr><lf>

Server Address: 1 (0x01), Function Number: 15 (0x0F)
Function Name : Write Multiple Coils

Starting Address: 0x0014, Quantity of Coils: 10 (0x000A)
                                                        (Request Event Number: 0x00000000002C)
<------------------------------------------------------------------------------------------------
```

Function 16

```
---- [Normal] [Request] (0x00000000002E) (xxxx/xx/xx xx:xx:xx) ---------------------------------->
:0110000100020400 0A0102DB<cr><lf>

Server Address: 1 (0x01), Function Number: 16 (0x10)
Function Name : Write Multiple Registers

Starting Address: 0x0002, Quantity of Registers: 2 (0x0002)
Byte Count      : 4 (0x04)
Register Value  : ----- Following --------------------
0x0002:    10 (0x000A), 0x0003:   258 (0x0102)
------------------------------------------------------------------------------------------------>

<--- [Normal] [Response] (0x00000000004A) (xxxx/xx/xx xx:xx:xx) ----------------------------------
:011000010002EC<cr><lf>

Server Address: 1 (0x01), Function Number: 16 (0x10)
Function Name : Write Multiple Registers

Starting Address: 0x0002, Quantity of Registers: 2 (0x0002)
                                                        (Request Event Number: 0x00000000002E)
<------------------------------------------------------------------------------------------------
```

Function 17

```
---- [Normal] [Request] (0x00000000001D) (xxxx/xx/xx xx:xx:xx) ---------------------------------->
:1111DE<cr><lf>

Server Address: 17 (0x11), Function Number: 17 (0x11)
Function Name : Report Server ID
------------------------------------------------------------------------------------------------>

<--- [Normal] [Response] (0x000000000027) (xxxx/xx/xx xx:xx:xx) ----------------------------------
:1111050123FF45670A<cr><lf>

Server Address: 17 (0x11), Function Number: 17 (0x11)
Function Name : Report Server ID

Byte Count: 5 (0x05)
From Server ID to Additional Data: ----- Following --------------------
  1 (0x01),  35 (0x23), 255 (0xFF),  69 (0x45), 103 (0x67)
                                                        (Request Event Number: 0x00000000001D)
<------------------------------------------------------------------------------------------------
```

Function 20

```
---- [Normal] [Request] (0x00000000002F) (xxxx/xx/xx xx:xx:xx) ---------------------------------->
:01140E060004000100020600030009000 2BC<cr><lf>

Server Address: 1 (0x01), Function Number: 20 (0x14)
Function Name : Read File Record

Byte Count  : 14 (0x0E)
Group Number: 1 (0x0001), Reference Type: 6 (0x06)
File Number : 4 (0x0004), Starting Record: 1 (0x0001), Length of Record: 2 (0x0002)
Group Number: 2 (0x0002), Reference Type: 6 (0x06)
File Number : 3 (0x0003), Starting Record: 9 (0x0009), Length of Record: 2 (0x0002)
------------------------------------------------------------------------------------------------>

<--- [Normal] [Response] (0x000000000056) (xxxx/xx/xx xx:xx:xx) ----------------------------------
:01140C05060DFE0020050633CD00405E<cr><lf>

Server Address: 1 (0x01), Function Number: 20 (0x14)
Function Name : Read File Record

Response Data Length: 12 (0x0C)
Group Number: 1 (0x0001), Reference Type: 6 (0x06), Byte Count: 5 (0x05)
File Number : 4 (0x0004), Starting Record: 1 (0x0001), Length of Record: 2 (0x0002)
0x0001:  3582 (0x0DFE), 0x0002:    32 (0x0020)
Group Number: 2 (0x0002), Reference Type: 6 (0x06), Byte Count: 5 (0x05)
File Number : 3 (0x0003), Starting Record: 9 (0x0009), Length of Record: 2 (0x0002)
0x0009: 13261 (0x33CD), 0x000A:    64 (0x0040)
                                                                (Request Event Number: 0x00000000002F)
<------------------------------------------------------------------------------------------------
```

Function 21

```
---- [Normal] [Request] (0x00000000003D) (xxxx/xx/xx xx:xx:xx) ---------------------------------->
:01150D0600040007000306AF04BE100D35<cr><lf>

Server Address: 1 (0x01), Function Number: 21 (0x15)
Function Name : Write File Record

Request Data Length: 13 (0x0D)
Group Number: 1 (0x0016), Reference Type: 6 (0x06)
File Number : 4 (0x0004), Starting Record: 7 (0x0007), Length of Record: 3 (0x0003)
0x0007:  1711 (0x06AF), 0x0008:  1214 (0x04BE), 0x0009:  4109 (0x100D)
------------------------------------------------------------------------------------------------>

<--- [Normal] [Response] (0x000000000063) (xxxx/xx/xx xx:xx:xx) ----------------------------------
:01150D0600040007000306AF04BE100D35<cr><lf>

Server Address: 1 (0x01), Function Number: 21 (0x15)
Function Name : Write File Record

Response Data Length: 13 (0x0D)
Group Number: 1 (0x0016), Reference Type: 6 (0x06)
File Number : 4 (0x0004), Starting Record: 7 (0x0007), Length of Record: 3 (0x0003)
0x0007:  1711 (0x06AF), 0x0008:  1214 (0x04BE), 0x0009:  4109 (0x100D)
                                                                (Request Event Number: 0x00000000003D)
<------------------------------------------------------------------------------------------------
```

Function 22

```
---- [Normal] [Request] (0x000000000030) (xxxx/xx/xx xx:xx:xx) ---------------------------------->
:1116000400F20025BE<cr><lf>

Server Address: 17 (0x11), Function Number: 22 (0x16)
Function Name : Mask Write Register

Reference Address: 0x0005, AND Mask: 242 (0x00F2), OR Mask 37 (0x0025)
------------------------------------------------------------------------------------------------->

<--- [Normal] [Response] (0x000000000045) (xxxx/xx/xx xx:xx:xx) ----------------------------------
:1116000400F20025BE<cr><lf>

Server Address: 17 (0x11), Function Number: 22 (0x16)
Function Name : Mask Write Register

Reference Address: 0x0005, AND Mask: 242 (0x00F2), OR Mask 37 (0x0025)
                                                        (Request Event Number: 0x000000000030)
<-------------------------------------------------------------------------------------------------
```

Function 23

```
---- [Normal] [Request] (0x000000000035) (xxxx/xx/xx xx:xx:xx) ---------------------------------->
:011700030006000E00030600FF00FF00FFCB<cr><lf>

Server Address: 1 (0x01), Function Number: 23 (0x17)
Function Name : Read/Write Multiple Registers

Read Starting Address : 0x0004, Quantity to Read : 6 (0x0006)
Write Starting Address: 0x000F, Quantity to Write: 3 (0x0003)
Byte Count    : 6 (0x06)
Register Value: ----- Following --------------------
0x000F:   255 (0x00FF), 0x0010:   255 (0x00FF), 0x0011:   255 (0x00FF)
------------------------------------------------------------------------------------------------->

<--- [Normal] [Response] (0x00000000005D) (xxxx/xx/xx xx:xx:xx) ----------------------------------
:01170C00FE0ACD00010003000D00FFF7<cr><lf>

Server Address: 1 (0x01), Function Number: 23 (0x17)
Function Name : Read/Write Multiple Registers

Byte Count    : 12 (0x0C)
Register Value: ----- Following --------------------
0x0004:   254 (0x00FE), 0x0005:  2765 (0x0ACD), 0x0006:     1 (0x0001), 0x0007:     3 (0x0003),
0x0008:    13 (0x000D), 0x0009:   255 (0x00FF)
                                                        (Request Event Number: 0x000000000035)
<-------------------------------------------------------------------------------------------------
```

Function 24

```
---- [Normal] [Request] (0x000000000029) (xxxx/xx/xx xx:xx:xx) ---------------------------------->
:011804DE05<cr><lf>

Server Address: 1 (0x01), Function Number: 24 (0x18)
Function Name : Read FIFO Queue

FIFO Pointer Address: 1246 (0x04DE)
------------------------------------------------------------------------------------------------->

<--- [Normal] [Response] (0x000000000037) (xxxx/xx/xx xx:xx:xx) ----------------------------------
:01180006000201B8128490<cr><lf>

Server Address: 1 (0x01), Function Number: 24 (0x18)
Function Name : Read FIFO Queue

Byte Count: 6 (0x0006), FIFO Count 2 (0x0002)
FIFO Value Register: ----- Following --------------------
0x04DE:   440 (0x01B8), 0x04DF:  4740 (0x1284)
                                                        (Request Event Number: 0x000000000029)
<-------------------------------------------------------------------------------------------------
```

Function 43-13

```
---- [Normal] [Request] (0x000000000001) (xxxx/xx/xx xx:xx:xx) ---------------------------------->
:112B0D123456789ABCDEF0123456789ABCDEF01234567833<cr><lf>

Server Address: 17 (0x11), Function Number: 43-13 (0x2B-0x0D)
Function Name : Encapsulated Interface Transport - CANopen General Ref. Req. and Res. PDU

MEI Type: 13 (0x0D)
MEI Type Specific Data: ----- Following --------------------
0x12, 0x34, 0x56, 0x78, 0x9A, 0xBC, 0xDE, 0xF0, 0x12, 0x34, 0x56, 0x78, 0x9A, 0xBC, 0xDE, 0xF0,
0x12, 0x34, 0x56, 0x78
------------------------------------------------------------------------------------------------>

<--- [Normal] [Response] (0x000000000035) (xxxx/xx/xx xx:xx:xx) ----------------------------------
:112B0D123456789ABCDEF0123456789ABCDEF01234567833<cr><lf>

Server Address: 17 (0x11), Function Number: 43-13 (0x2B-0x0D)
Function Name : Encapsulated Interface Transport - CANopen General Ref. Req. and Res. PDU

MEI Type: 13 (0x0D)
MEI Type Specific Data: ----- Following --------------------
0x12, 0x34, 0x56, 0x78, 0x9A, 0xBC, 0xDE, 0xF0, 0x12, 0x34, 0x56, 0x78, 0x9A, 0xBC, 0xDE, 0xF0,
0x12, 0x34, 0x56, 0x78
                                                          (Request Event Number: 0x000000000001)
<------------------------------------------------------------------------------------------------
```

Function 43-14

```
---- [Normal] [Request] (0x000000000003) (xxxx/xx/xx xx:xx:xx) ---------------------------------->
:012B0E0100C5<cr><lf>

Server Address: 1 (0x01), Function Number: 43-14 (0x2B-0x0E)
Function Name : Encapsulated Interface Transport - Read Device Identification

MEI Type: 14 (0x0E)
Device ID Code   : 1 (0x01) Request to get the basic device identification (Stream Access)
Object ID        : 0 (0x00) VendorName (Category: Basic)
------------------------------------------------------------------------------------------------>

<--- [Normal] [Response] (0x000000000015) (xxxx/xx/xx xx:xx:xx) ----------------------------------
:012B0E01010000030016436F6D70616E79206964656E74696669636174696F6E010D50726F6475637420636F64655802
 0556322E3131C9<cr><lf>

Server Address: 1 (0x01), Function Number: 43-14 (0x2B-0x0E)
Function Name : Encapsulated Interface Transport - Read Device Identification

MEI Type: 14 (0x0E)
Device ID Code   : 1 (0x01)  Request to get the basic device identification (Stream Access)
Conformity Level : 1 (0x01)  Basic identification (Stream access only)
More Follows     : 0 (0x00)  No more object are available.
Number Of Objects: 3 (0x03)
Object Data      : ----- Following --------------------
Object ID: 0 (0x00)  VendorName (Category: Basic)
  Object Length: 22 (0x16)
  0x43, 0x6F, 0x6D, 0x70, 0x61, 0x6E, 0x79, 0x20, 0x69, 0x64, 0x65, 0x6E, 0x74, 0x69, 0x66, 0x69,
  0x63, 0x61, 0x74, 0x69, 0x6F, 0x6E
  "Company identification"
Object ID: 1 (0x01)  ProductCode (Category: Basic)
  Object Length: 13 (0x0D)
  0x50, 0x72, 0x6F, 0x64, 0x75, 0x63, 0x74, 0x20, 0x63, 0x6F, 0x64, 0x65, 0x58
  "Product codeX"
Object ID: 2 (0x02)  MajorMinorRevision (Category: Basic)
  Object Length: 5 (0x05)
  0x56, 0x32, 0x2E, 0x31, 0x31
  "V2.11"
                                                          (Request Event Number: 0x000000000003)
<------------------------------------------------------------------------------------------------
```

Serial Port Monitor and Analyzer Series
Protocol Analyzer for Modbus ASCII (Model: AKM-RSM-FM0) User's Manual

November, 2016 Version 0.3.0.0
December, 2016 Version 0.4.0.0
January, 2017 Version 1.0.0.0
July, 2019 Version 1.1.0.0
November, 2019 Version 1.1.1.0
February, 2022 Version 1.2.0.0
July, 2023 Version 1.3.0.0

Publishing office: Akiyama Manufacturing