AP-IEB2

Protocol Analyze Unit For IEBus™

USER'S MANUAL

Version 1 2001. 10
Application Corporation

Contents

Page

1. Introduction

The AP-IEB2 is a development support tool that provides a monitoring function for networks constructed using IEBus as well as a dummy device function.
A function for detection of various events performs filtering and starting and stopping of monitoring.
The AP-IEB2 is also equipped to use the macro language employed in Application Corporation's AP-ALDM, enabling easy emulation of complex devices.

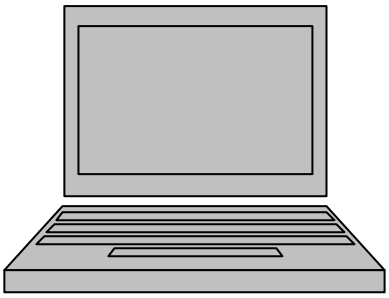
Frames can be easily viewed as they pass through the IEBus
Using the frame monitoring function, frames can not only be displayed but searched, saved and loaded as well. In addition, the save function can save previous AP-PAB–compatible formats and images in text format.

In addition to monitoring frames, the AP-IEB2 can send and receive frames, providing a function for testing frame sending and receiving in devices being tested.

The AP-IEB2 can also behave as a dummy device, using the advanced macro language provided.

2. Nomenclature

[1] Controlling PC

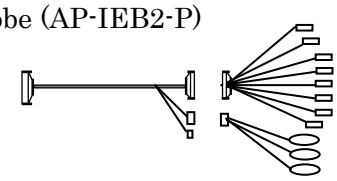


Required specifications : DOS/V(OADC specifications) Notebook PC with PC card slot
OS : Windows95/98/Me/2000

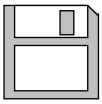
[2] AP-IEB2

AP-IEB2

Probe (AP-IEB2-P)



3.5-inch floppy disk



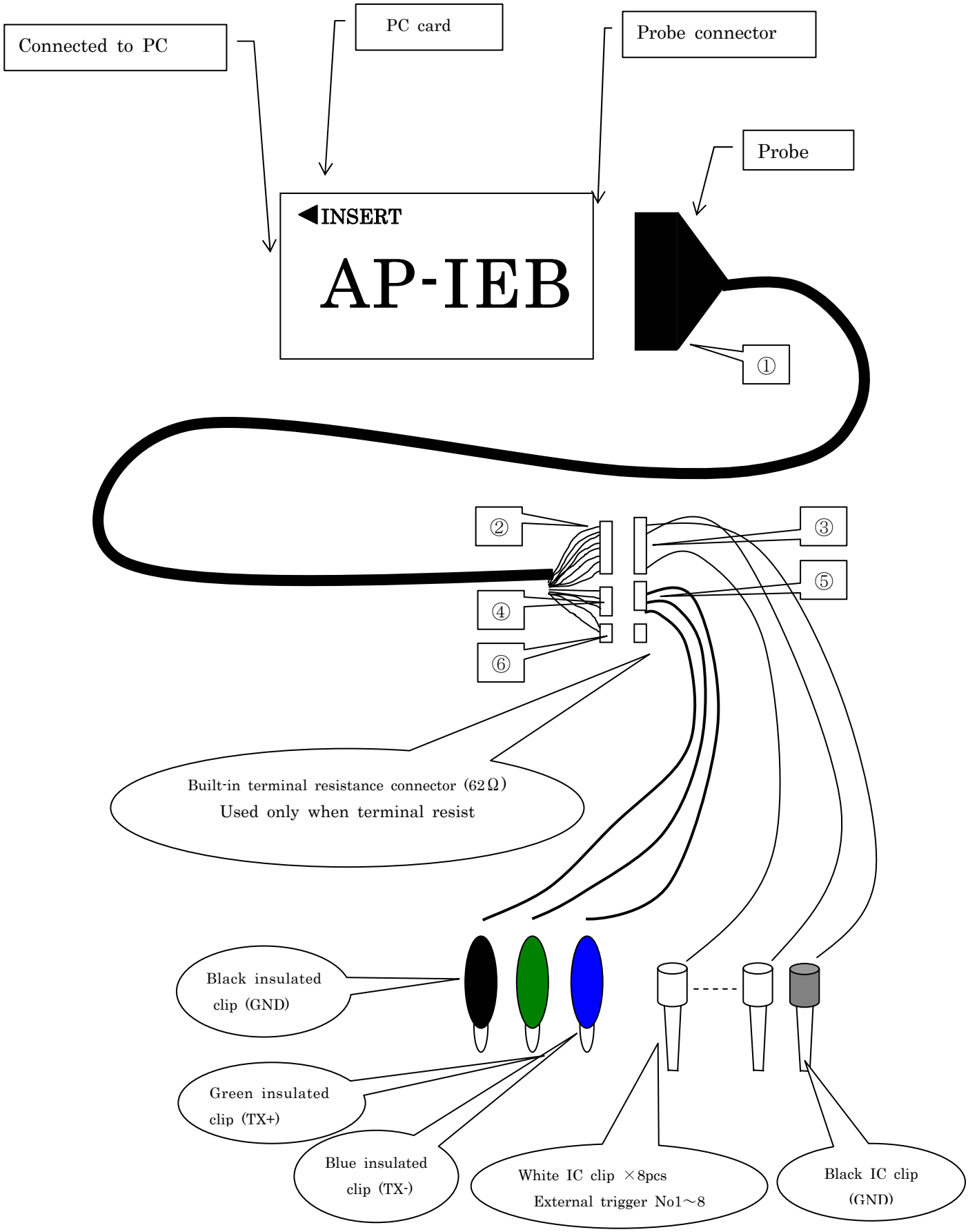
Terminal resistance connector
(62Ω terminal resistance)

Overall structural view

2.1. Specification

IEBus monitor function	: FPGA
IEBus emulation function	: uPD72042 (NEC) compatible with communication mode 1 and 0
Base clock	: 6.29MHz
Connection format	: Twisted-pair cable (approximately 1m)
TX+, TX-, GND	: Insulated clip
External trigger	: IC clip (Electrical specification: TTL input)
Interface	: TYPEⅡ PC card I/F(5V)
Host machine	: DOS/V
OS	: Windows95/98/Me/2000
Power supply	: Supplied from PC.
Current consumption	: Approximately 100mA
Dimensions	: 85.6×54.0×5.0(mm)

2.2. IEBus protocol analyzer unit: Description of Parts



PC card　　Connect the PC card to the PC.
In doing so, be careful of the following points.

- ✓ Do not exert pressure or place heavy objects on the PC card.
- ✓ Do not drop the PC card or expose it to vibration or shock.
- ✓ Do not pull from the cord part when removing the cable or PC card.
- ✓ Avoid using or storing the PC card in hot, humid or dusty locations or in place exposed to direct sunlight.
- ✓ Avoid using or storing the PC card in locations subject to sudden changes in temperature or humidity.
- ✓ Do not drop liquids on the PC card or its accessories.
- ✓ Do not mistakenly connect the connector to another PC card.

Blue insulated clip	Connect to TX- on the IEBus.
Green insulated clip	Connect to TX+ on the IEBus.
Black insulated clip	Connect to the target GND.
White IC clip	This is used for input of the external trigger signal. Input the TTL level signal here.
Black IC clip	When using an external trigger signal, connect this to the target GND.

2.3 Connector specifications

These numbers corresponds to the reference numbers on the previous page.

① Hirose: NX30TA-32PAA + NX32TA-CV1

Pin No.	Signal name	Pin No.	Signal name
1	GND	17	GND
2	External trigger 1	18	Reserved
3	GND	19	GND
4	External trigger 2	20	Reserved
5	GND	21	TX+
6	External trigger 3	22	TX−
7	GND	23	N.C.
8	External trigger 4	24	Reserved
9	GND	25	N.C.
10	External trigger 5	26	Reserved
11	GND	27	GND
12	External trigger 6	28	Reserved
13	GND	29	GND
14	External trigger 7	30	Reserved
15	GND	31	TX+
16	External trigger 8	32	TX−

Do not use pins marked “reserved” or “N.C.”.

② Hirose: Df1E-10S-2.5C
③ Hirose: DF1E-10EP-2.5C

Pin No.	Signal name
1	External trigger 1
2	External trigger 2
3	External trigger 3
4	External trigger 4
5	External trigger 5
6	External trigger 6
7	External trigger 7
8	External trigger 8
9	GND
10	N.C.

④ Hirose: DF1E-3S-2.5C
⑤ Hirose: DF1E-3EP-2.5C

Pin No.	Signal name
1	TX−
2	TX+
3	GND

⑥ Hirose: DF1E-2S-2.5C

Pin No.	Signal name
1	TX−
2	TX+

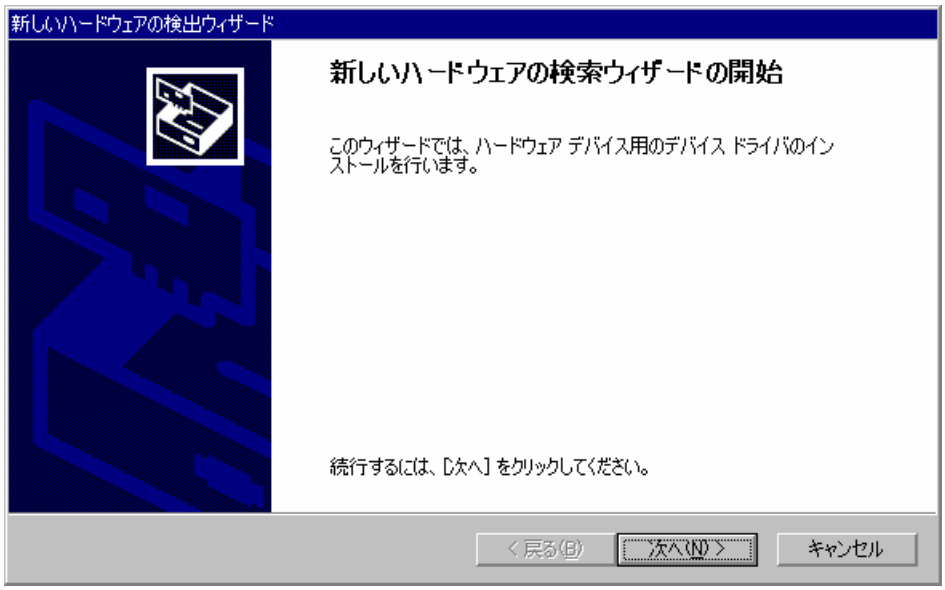
3. Installation

Please note that the procedure for the PC card depends on the OS used by the PC.

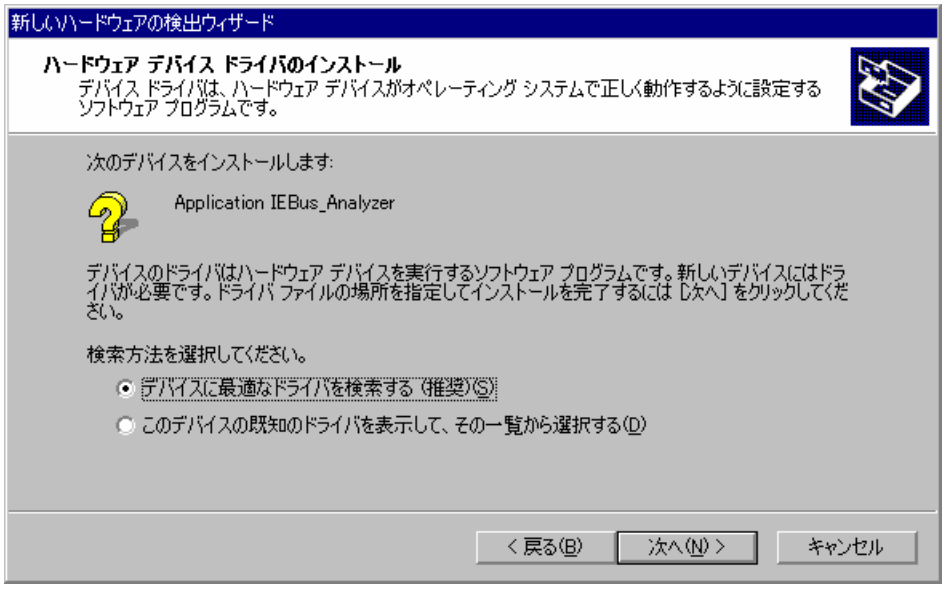
3.1 Installing the PC card (windows 2000)

Use the following procedure to install the PC card in the AP-IEB2.

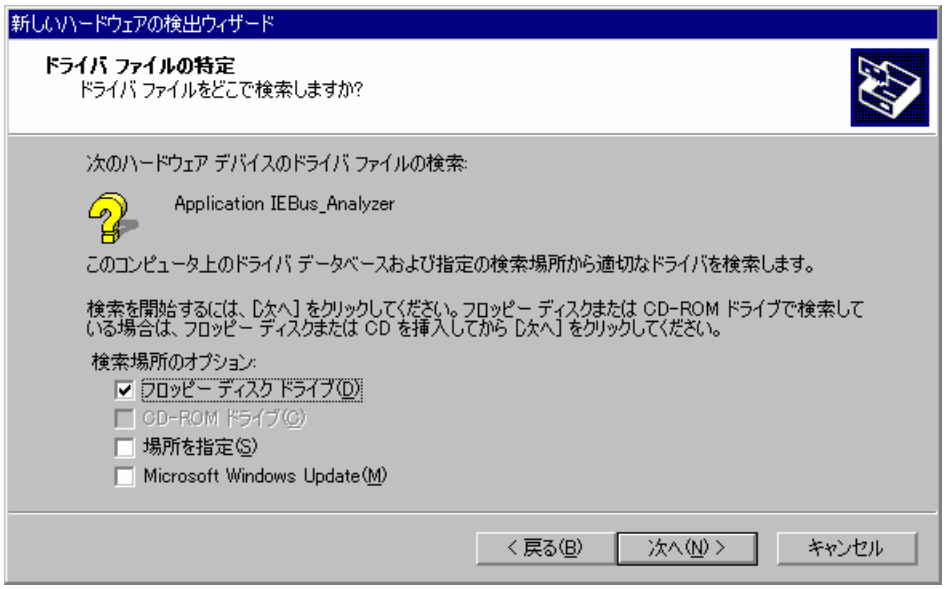
① Start Windows 2000 and log in using your administrator privileges.
② Insert the AP-IEB2 Setup Disk in the floppy drive.
③ Insert the AP-IEB2 in the PC card slot of the PC.
Refer to the manual of the PC for the correct method of installation.
④ The AP-IEB2 will be recognized automatically and the following window will appear.
Click Next.



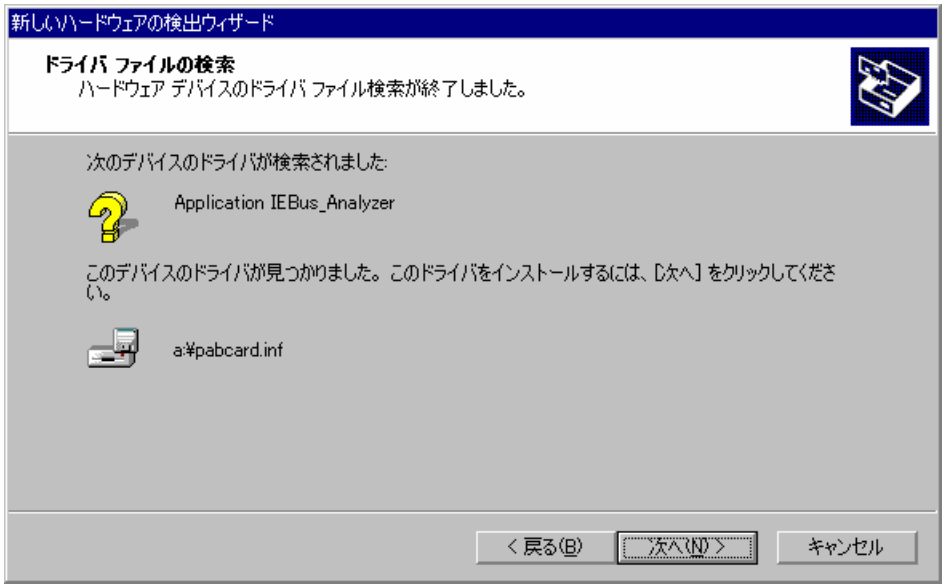
⑤ Click “Search for the best driver for this device” and click Next.



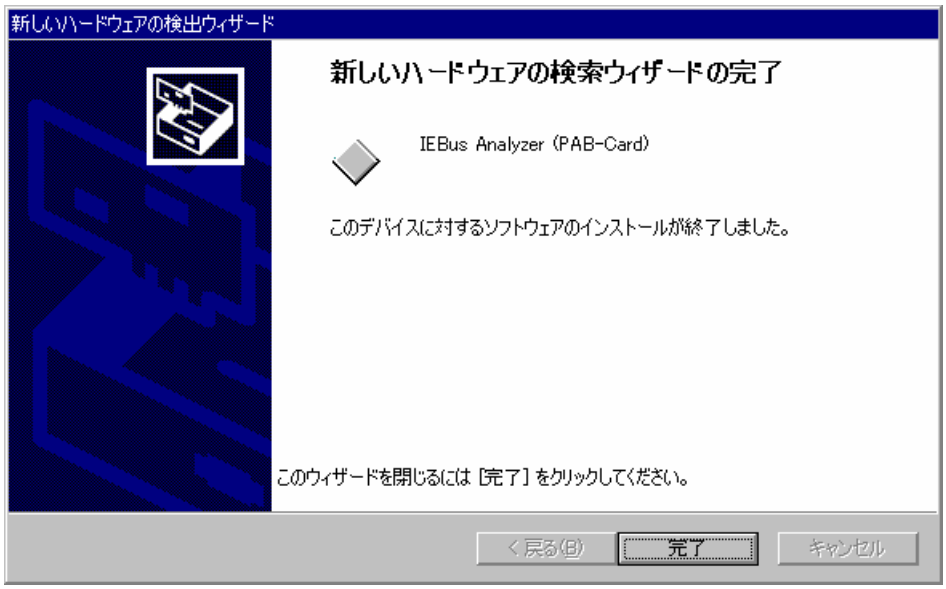
⑥ Set “Options for search location” as follows and click Next.



⑦ Verify that ‘pabcard. inf’ is displayed and click Next.



⑧ When the window below appears, click Finish.



3.2 Installing the PC card (windows98/Me)

Install the PC card in the AP-IEB2 as follows.
The windows shown below are from Windows98; windows in Windows Me may appear different

① Insert the AP-Ieb2 Setup Disk in the floppy drive.
② Insert the AP-IEB2 in the PC card slot of the PC.
For information on installation, please refer to the manual included with your PC.
③ The AP-IEB2 will be recognized automatically and the following window will Appear.
Click Next.

④ Click “Search for the best driver for the device in use” and click Next.

⑤ Select “floppy disk drive” and click Next.

⑥ Verify that ‘PABCARD INF’ is displayed and click Next.

⑦ If a message such as “PABCARD. SYS not found” appears, click “Floppy drive” and click OK.

⑧ When the windows below appears, click Finish.

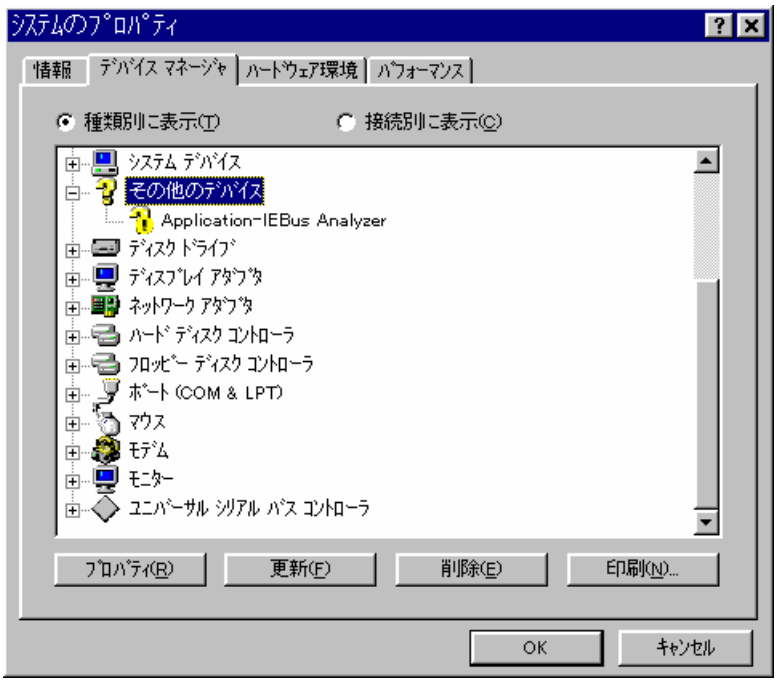
3.3 Installing the PC card (windows95)

Install the AP-IEB2 in the PC as follows.

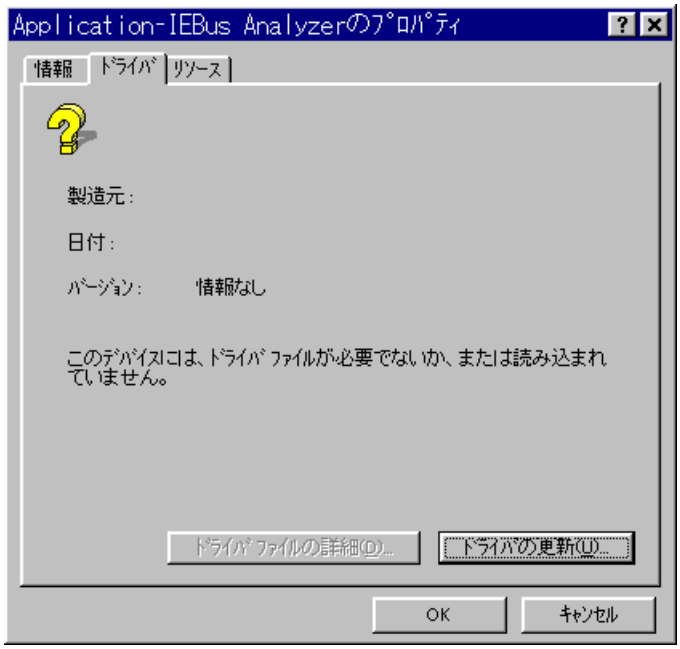
① Verify that the AP-IEB2 Setup disk is not in the floppy drive.
② Insert the AP-IEB2 in the PC card slot of the PC.
For information on installation, please refer to the manual included with your PC.
③ The AP-IEB2 will be recognized automatically and the following window will appear.
Click Next.

④ “Driver for this device not found” appears. Click Finish.

⑤ Select Start – Setting – Control Panel.
⑥ Double-click the System icon.
⑦ Double-click Hardware, that click the Device Manager tab.
⑧ Click Other devices to display properties for Application-IEBus Analyzer.



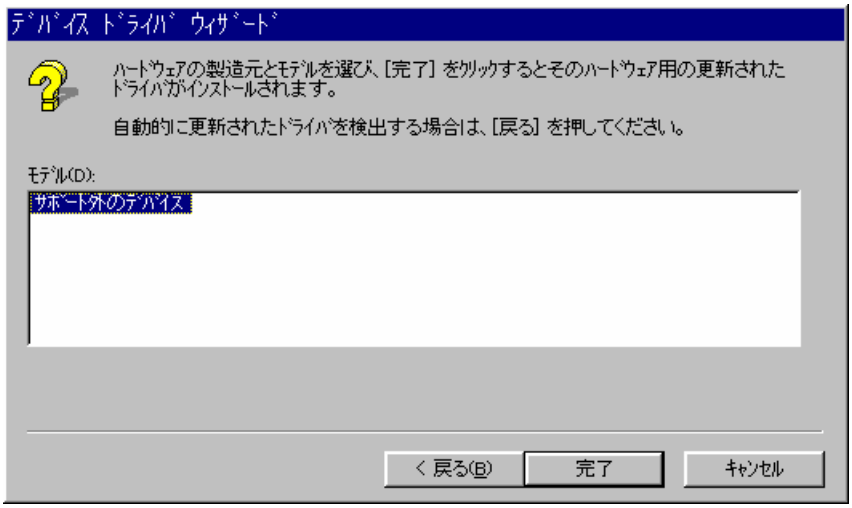
⑨ Double-click Drivers and click Change driver.



⑩ Select Choose driver from the list and click Next.

⑪ Select Other device and click Next.

⑫ Select Unsupported device and click Finish.



⑬ Click OK to close the Properties window for Application-IEBus Analyzer.
⑭ Verify that "!" does not appear beside Unsupported device and display Properties.

⑮ Click the Resource tab to display the I/O port address. You should make a note of this value (260 in the window displayed below), because you will need either it in the Initialize window to start AP-IEB2

3.4 Installing the program

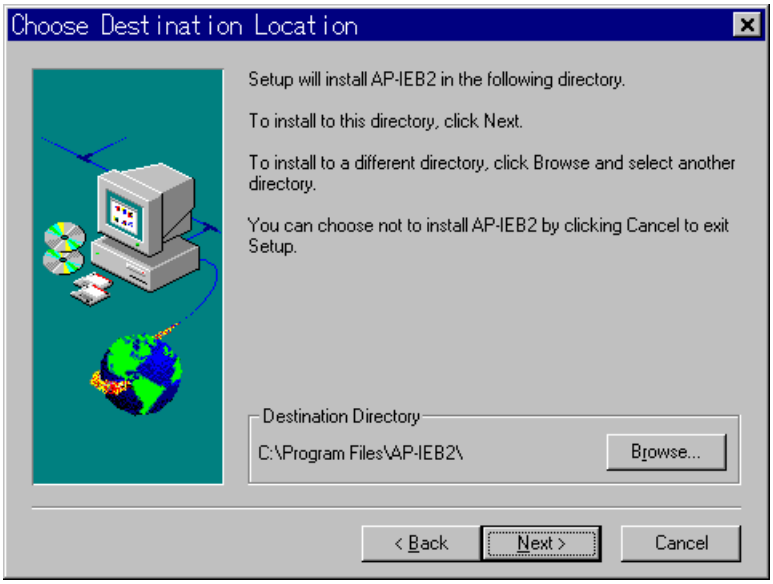
Install the program as follows.

> Note: Because the program file is stored on the disk in compressed form, you cannot use it simply by copying it to your hard disk. Use the setup program to install the program correctly.

(1) Start the setup program.
Insert the AP-IEB2 Setup Disk in the floppy drive and the execute "SETUP.EXE"

(2) Start installation

To start installation, click Next. To quit the installation procedure, click Cancel.



Specify the folder you wish to install the program in. If you are satisfied with the default installation folder, click Next. To change the installation folder, click Browse… and specify the folder you wish to install the program in.

(4) Setup is complete

The files needed to execute AP-IEB2 have been copied. Click OK. The setup process is complete.

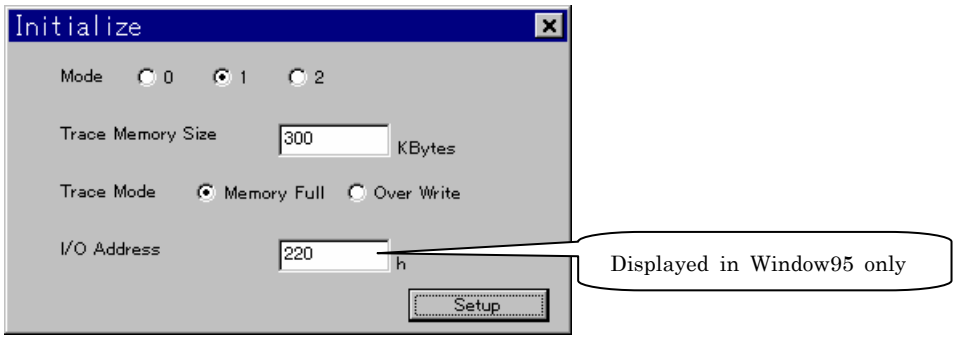
The next time you wish to execute AP-IEB2, simply click the AP-IEB2 icon from the group menu created in the installation process.

4. Initialize window

(1) Function

This window always appears when the program is launched.
It is used to set up initial hardware and software setting.

(2) Window



(3) Operation (items can be entered in the window or selected using the mouse)

① Select mode

Set the IEBus communication mode.
0:Mode0 1:Mode1 2:Mode2

② Enter trace memory size

Enter the memory size in Kbyte (Max 30720KByte)
The specified memory size is stored in your PC's memory.
One frame consumes about 526 bytes.

③ Select trace mode

Memory Full

Operations will be stopped if the trace memory size is exceeded.

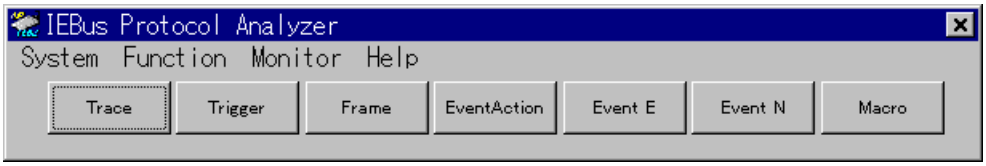
Over Write

When the selected trace memory size is exceeded, the record stored at the first index number is deleted and overwritten by the new data.

④ Enter I/O Address

This item appears if Windows95 is used.
Use the control panel to look up and enter the I/O address allocated to the PC card.
In Windows98/Me/2000 the I/O address is allocated automatically, so this item does not appear.

When you are satisfied with the input or selection, click Setup to display the main menu screen.



5. Monitor Menu

Click Monitor in the menu bar of the main menu to display a series of submenu items.

Click the required items using the mouse to display them as necessary.

Submenu items

① Trigger Conditions
Same as Trigger button on the main menu.

② Frame Conditions
Same as Frame button on the main menu.

③ Event Action
Same as Event Action button on the main menu.

④ External Trigger Setting
Same as Event E button on the main menu.

⑤ Noise Setting
Same as Event N button on the main menu.

⑥ Trace
Same as Traced button on the main menu.

5.1. Trace

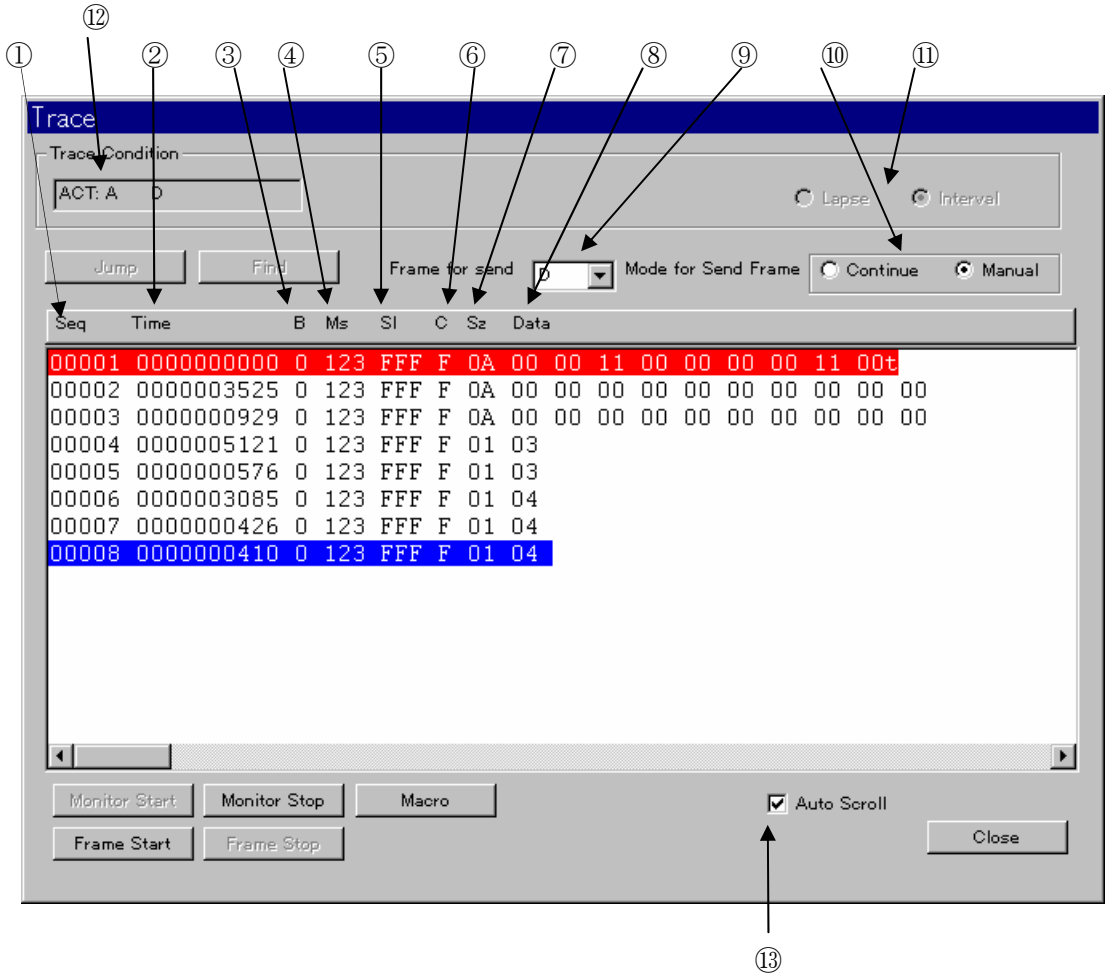
(1) Function

This function traces communication frames to display them in real time.
Communication frames can be sent by selecting the frame number of the frame to be sent.

Description of operation buttons

Monitor Start	: Starts the monitor.
Monitor Stop	: Stops the monitor.
Frame Start	: Starts frame transmission.
Frame Stop	: Stops frame transmission.
Macro	: Opens the macro window.
Close	: Closes the trace window and returns to the main menu.
Jump	: Jumps to the top or last display line after monitor is stopped.
Find	: Searches frames after monitor is stopped.

(2) Window



The cursor is displayed in blue.
Frames in which errors occur are displayed in red.

t : Timing error
p : Parity error
a : Acknowledge error

(3) Description of items displayed in the window

① Seq	Data sequence number. Numbers are displayed from 1 to 30000; the display reverts to 00000.
② Time	Indicates elapsed frame time (in 1msec units). This item is displayed in the mode set in ⑪ Laps/ Interval.
③ B	Multiple-address bit value. 0: Multiple address 1: Normal
④ Ms	Master address value.
⑤ Sl	Slave address value.
⑥ C	Control bit value.
⑦ Sz	Data size.
⑧ Data	Data value.
⑨ Frame for Send	Indicates the frame sent when the Frame Start button is clicked.
⑩ Mode for Send Frame	When Continue is specified for the frame sent when the Frame Start button is clicked, and Send Frame [Continue] is selected, the number of frames and send time set in Frame Conditions is followed. When Manual is selected, only one frame is sent.
⑪ Laps/Interval	When laps is selected, cumulative display mode is used in ②Time. When Interval is selected, interval display mode is used in ② Time.
⑫ ACT	Displays conditions of enables event actions. A : ····· Monitor Start B : ····· Monitor Stop C : ····· Qualify D : ····· Noise E : ····· Highlight
⑬ Auto Scroll	When this item is checked, the window scrolls when the monitor data in the list box is full, so that the newest data is always displayed. When this item is unchecked, data scrolling stops at that point. Checking and unchecking is available during monitoring.

(4) Operation

[1] Starting the monitor

When the Monitor Start button is clicked, monitoring starts using the conditions set in Event Action.

When monitoring starts, communication frames in the IEBus are traced, stored in trace memory and displayed in real time.
These items are written according to the conditions set in the Memory Start window.
(Memory full、Over Write 、Trace Memory Size)

Memory Full :Monitoring stops because trace memory size is exceeded.
Over Write :Old data is overwritten when memory size is exceeded, and data continues to be added.

〔Note〕 If the IEBus communication interval is short and the PC is not able to process data with sufficient speed, overflow in the AP-IEB2 buffer may occur. In these cases a message is displayed and monitoring stops. If this occurs, it is recommended that you use a faster PC and/or quit other applications running at the same time.

[2] Stopping the monitor

When Monitor Stop is ON in Event Action

When trigger conditions are established during monitoring, a message is displayed to indicate that trigger conditions are established and the monitor stops automatically.

To stop monitoring, click Monitor Stop.

[3] Frame transmission

① Using the mouse, select frame transmission mode.
Click the Mode- Continue- Manual button using the mouse.

Continue	Frames are sent at an interval of 0-9999msrc, 1-9999 times or an unlimited number of times.
Manual	Frames are sent one at a time.

② To send frames, click the Frame Start button.

[4] Stopping frame transmission

Frame transmission can be stopped as follows.

Continue	Transmission is stopped after a specified number of frames are sent.
Manual	Transmission is stopped after a single frame is sent.

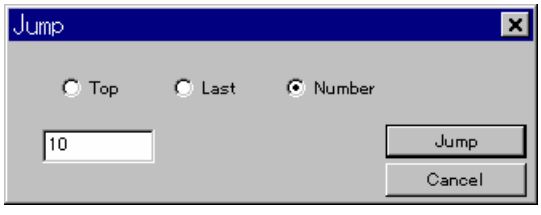
To stop frame transmission at any time, click Frame Stop.

[5] Executing macros

To open the Macro window, click the Macro button.

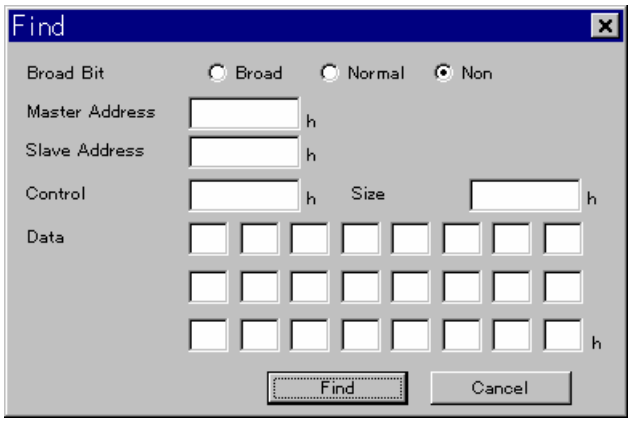
[6] Jump

Top	Moves to the first transmission frame in trace memory.
Last	Moves to the last transmission frame in trace memory.
Number	Moves to the frame whose sequence number is specified.



[7] Find

Enter search conditions to move to the frame that matches those conditions.
Items where nothing is entered are not included in the search conditions.

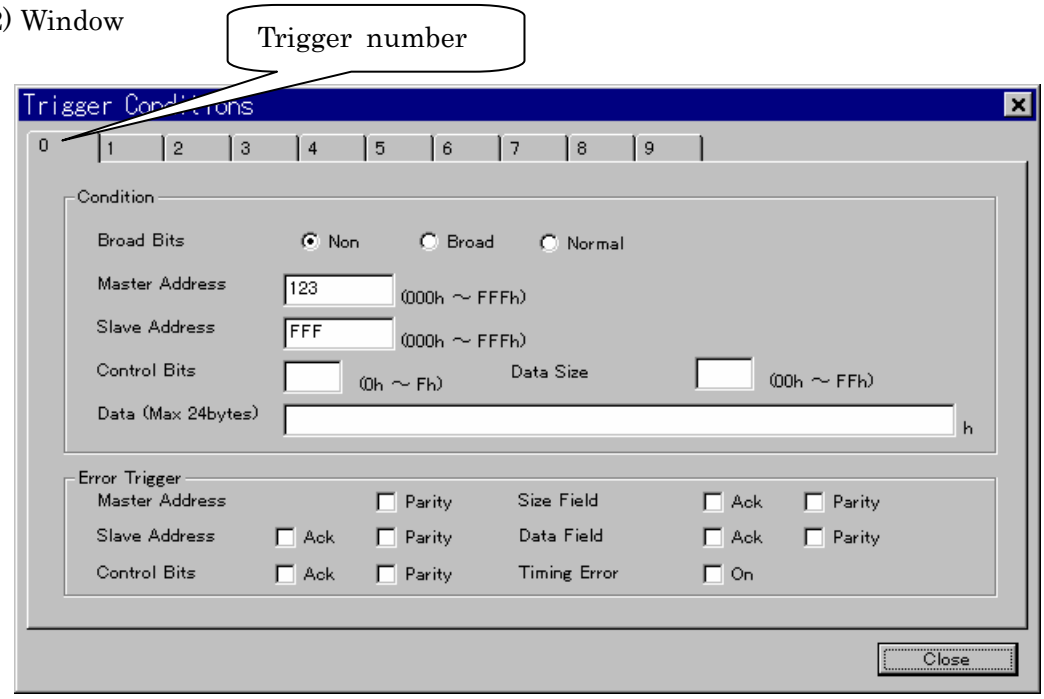


5.2. Trigger Conditions

(1) Function

This item sets the trigger conditions used in event action.

(2) Window



(3) Description

All of the trigger conditions ① through ⑥ are established. (AMD applies to each item)

The trigger condition setting window uses a tablet format. To change trigger conditions, click items 0 through 9.

Items not set appear blank.

① Broad Bits selection

Non : Multiple-address bits are ignored.
Broad : Used for multiple-address communication.
Normal : Used for individual communication.

② Master Address input

A hexadecimal number can be entered from 000 to FFF.

③ Slave Address input

A hexadecimal number can be entered from 000 to FFF.

④ Control Bits

A hexadecimal number can be entered from 0 to F.

⑤ Data input

A hexadecimal number can be entered from 000 to FFF.
Up to 24Byte can be entered.

⑥ Error Trigger (communication error) check box input

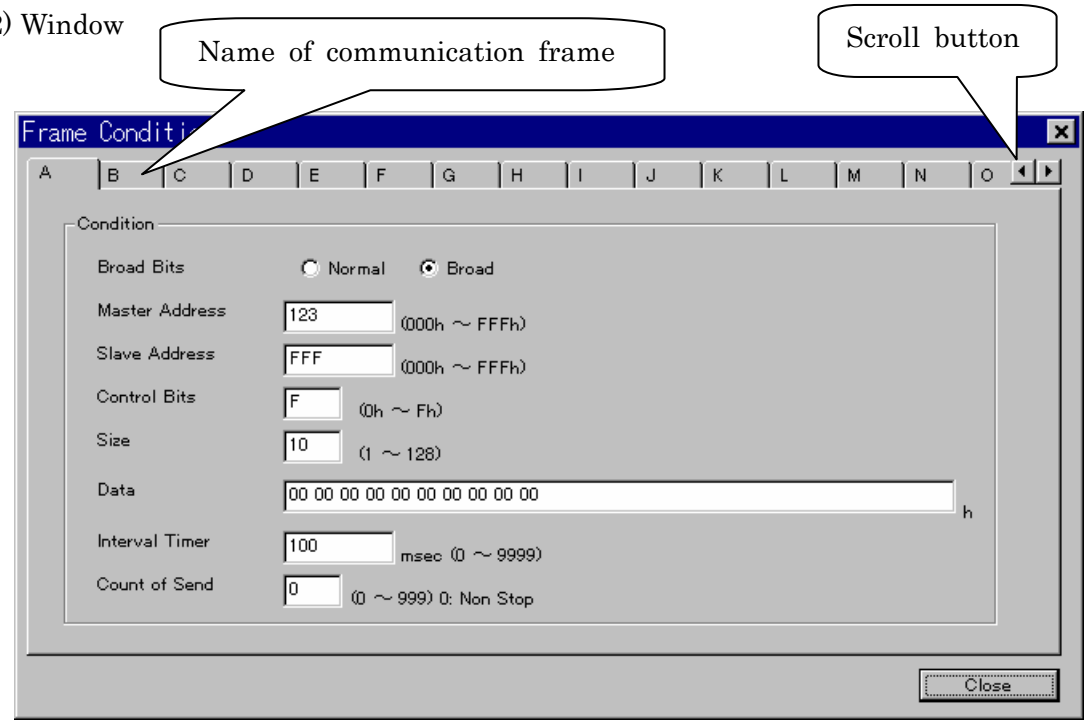
When multiple check boxes are checked ON, the OR condition applies.
If all check boxes are OFF, no conditions are set.
Ack :An acknowledgment error has occurred.
Parity :A parity error has occurred.
Timing Error :A timing error has occurred.

5.3. Frame Conditions (creating frames)

(1) Function

This item is used to create communication frames for sending, Frame values, transmission interval and number of transmissions can be set.

(2) Window



(3) Description

The registrations procedure is as follows.

① Up to 26 communication frames from A to Z can be created.
Select a frame by clicking one of the letters A to Z on the tablet
Because the tablet cannot display all letters at once, click the scroll button to scroll through the letters

② Selecting multiple-address bits using the mouse.
Broad :Used for multiple-address communication.
Normal :Used for individual communication.

③ Master address input
A hexadecimal number can be entered from 000 to FFF.

④ Slave address input
A hexadecimal number can be entered from 000 to FFF.

⑤ Control bit input
A hexadecimal number can be entered from 0 to F.

⑥ Size input
Size can be entered in units of 1 Byte, from 1 to 128.

⑦ Data input
Enter the data for size input in ⑥ in hexadecimal form.

⑧ Interval Timer
Enter the communication interval in milliseconds.
If 0 is entered, an interval of several milliseconds is registered.

⑨ Send count

Enter the number of frames sent in decimal form.
Enter a number from 0 to 999.
If 0 is set, the number of transmissions is unlimited.

⑩ Click the Close button to register the data in each frame.

［Note］

Relationship between size and data in frame transmission

・Size is smaller than data・・・Precedence is given to size.
・Size is larger than data・・・・00H is sent where data has insufficient size.

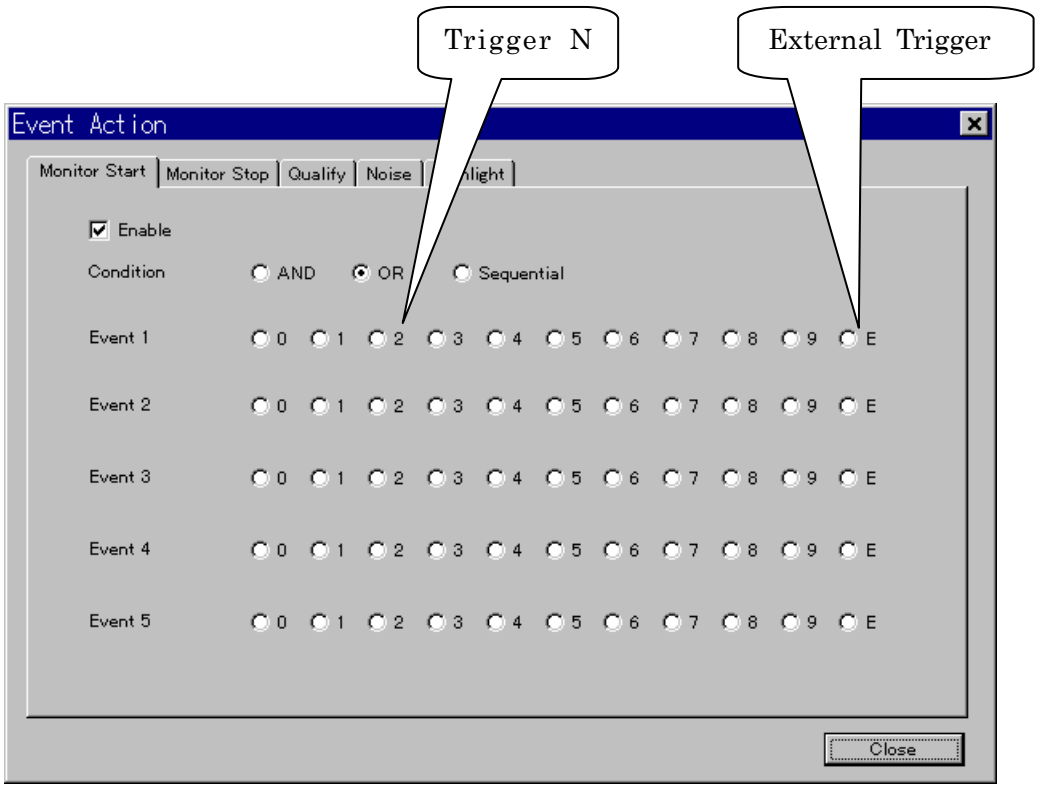
5.4. Event Action (setting action conditions)

(1) Monitor Start

Monitoring starts when the event set here is established.

To use this action, check the Enable box.

To start monitoring with no conditions set, simply check the Enable box with no event set.



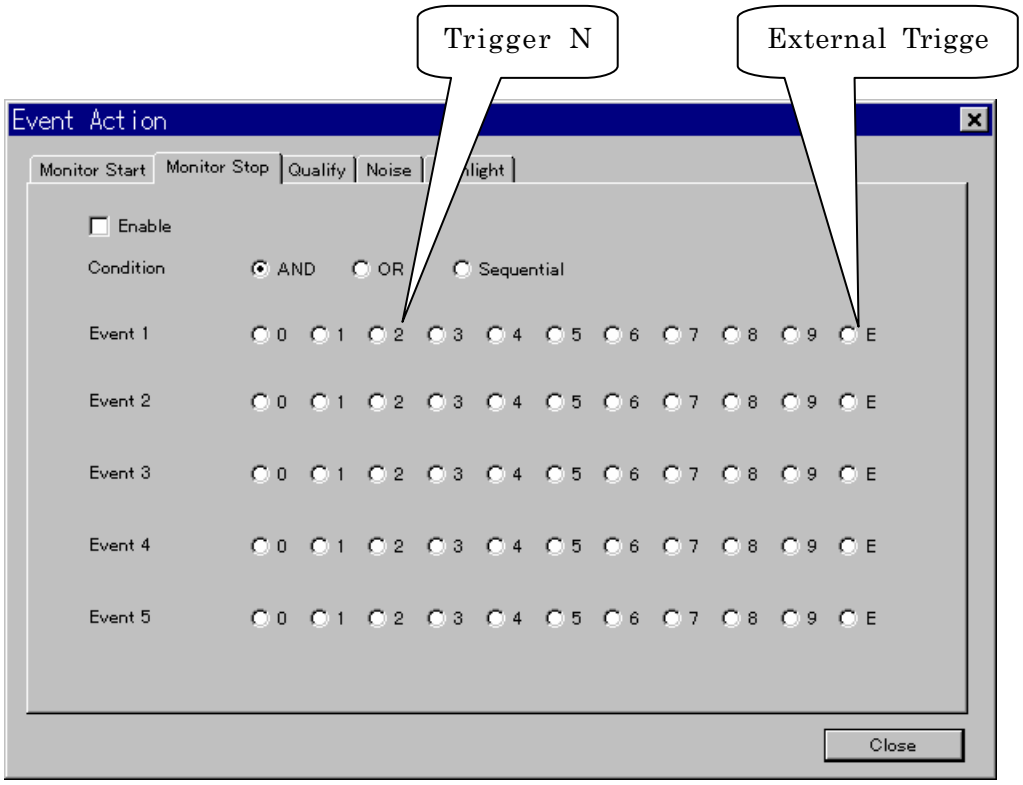
・ Condition

AND	With Event 1 set to 0, Event 2 set to 1 and Event 3 set to 2, monitoring starts when all three events occur regardless of order.
OR	With Event 1 set to 0, Event 2 set to 1 and Event 3 set to 2, monitoring starts when any of the three events occur regardless of order.
Sequential	With Event 1 set to 0, Event 2 set to 1 and Event 3 set to 2, monitoring starts when these three events occur in order.

(2) Monitor Stop

Monitoring stops when the event set here is established.

To use this action, check the Enable box.



・Condition

AND	With Event 1 set to 0, Event 2 set to 1 and Event 3 set to 2, monitoring stops when all three events occur regardless of order.
OR	With Event 1 set to 0, Event 2 set to 1 and Event 3 set to 2, monitoring stops when any of the three events occurs regardless of order.
Sequential	With Event 1 set to 0, Event 2 set to 1 and Event 3 set to 2, monitoring stops when these three events occur in order.

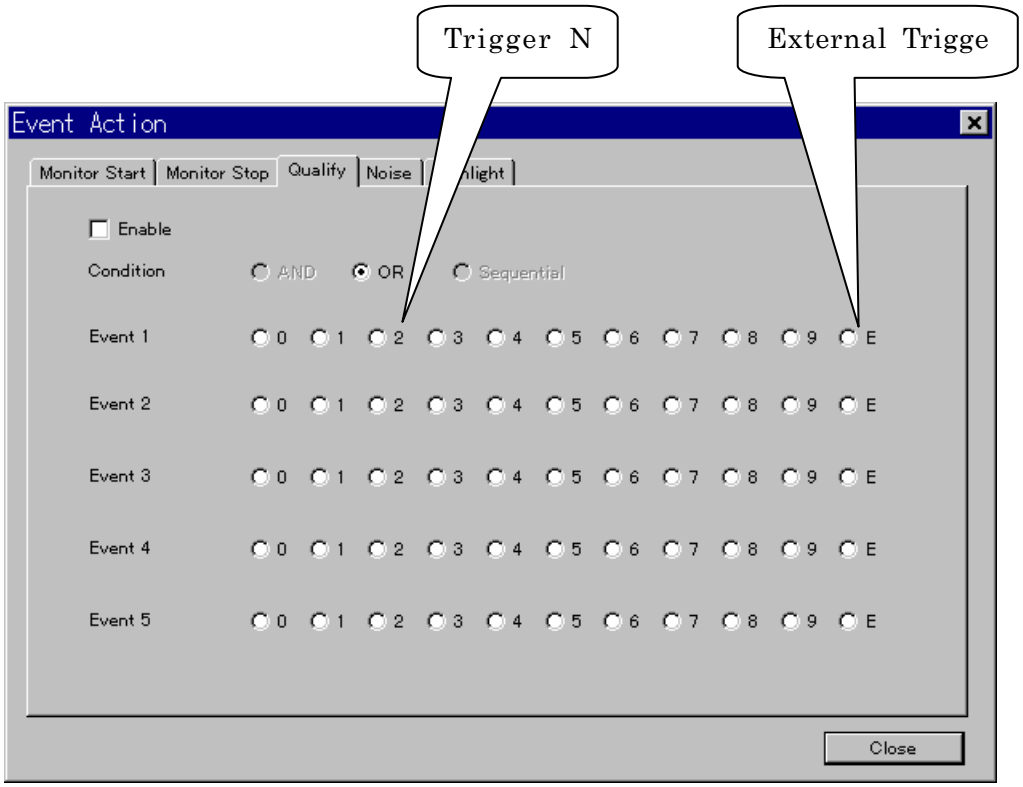
(3) Qualify

Monitoring of communication frames only is performed when the events set here are established.

Monitoring is performed even if no monitor start conditions are established and monitoring is not started.

To use this action, click the Enable box.

Because only the OR condition is available for this item, monitoring is performed regardless of the order of events set.

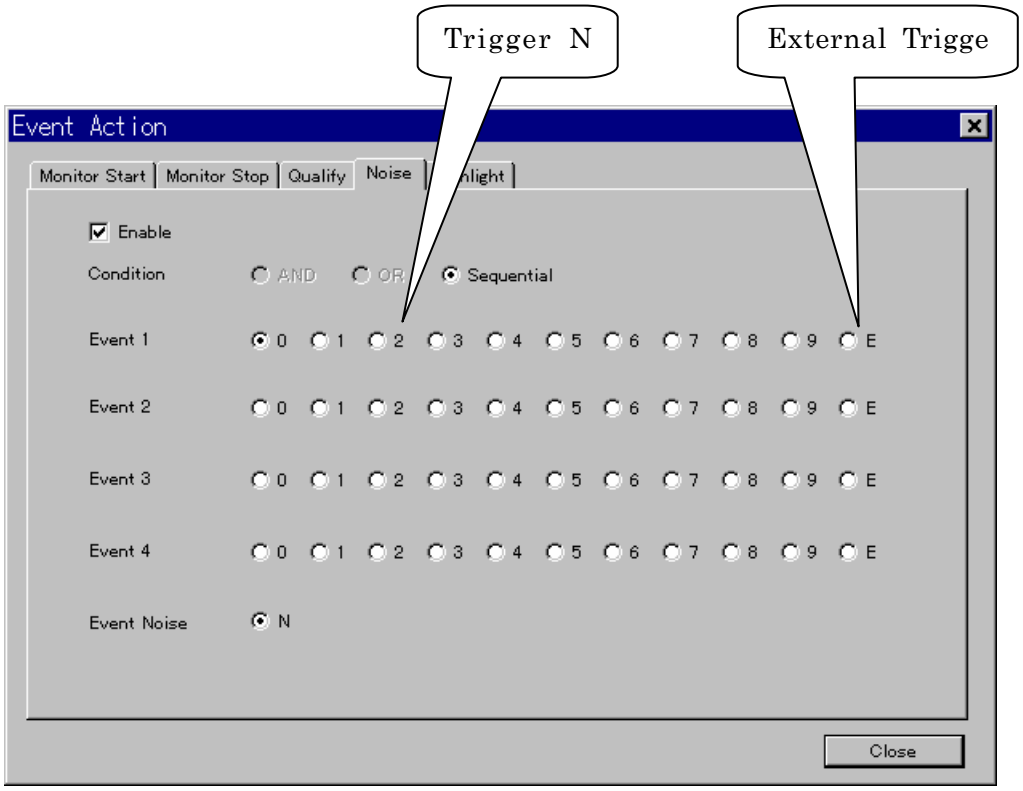


(4) Noise

Noise set in Event N is generated for the frame after the communication frame in which the events set here are established.

To use this action, click the Enable box.

Only the Sequential condition is available for this item.



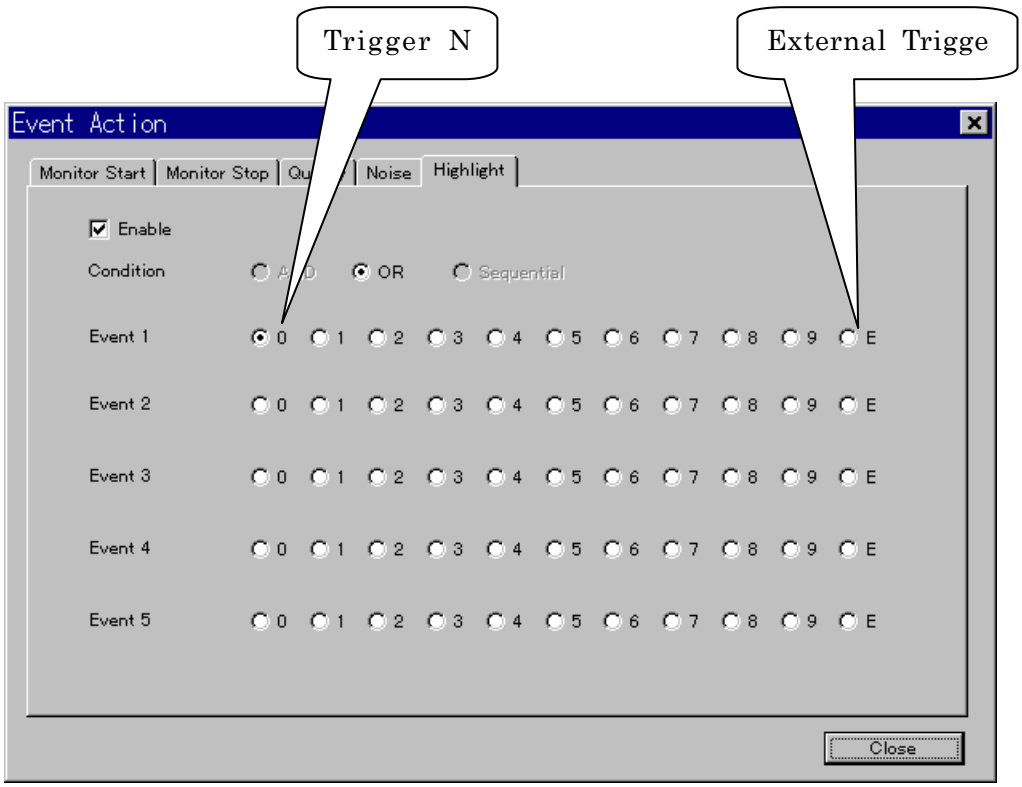
In the above trigger setting, after trigger 0 occurs, AP-IEB2 waits for detection of Even N.

(6) Highlight

Communication frames in which the events set here are established are highlighted in yellow-green.

To use this action, click the Enable box.

Only the OR condition is available for this item. Events set here are highlighted in yellow-green regardless of order.

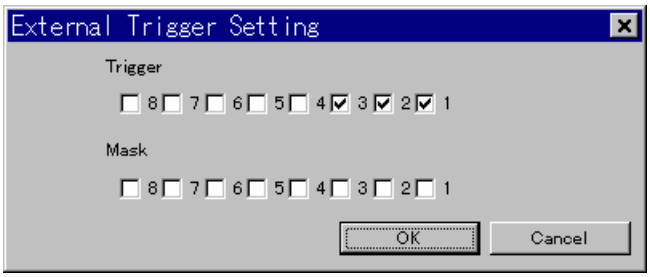


5.5. Event E (External Trigger Setting: External trigger setting)

Event can be set using an external trigger terminal.

Trigger Set the signal pattern entered in bits 1 to 8.
Checking bits are set to High (TTL level).
Unchecked bits are set to Low.

Mask Content set in TRG can be masked in bit units.
Checked bits are masked.
Unchecked bit are non-masked.



In the above settings, even E occurs because Mask is not checked:

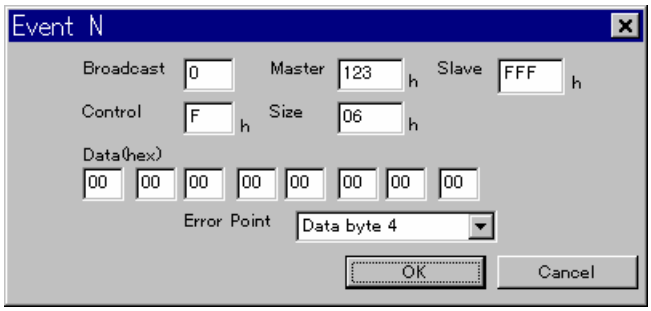
Trigger terminal no.	Status
1	High
2	High
3	High
4	Low
5	Low
6	Low
7	Low
8	Low

5.6. Event N (Noise setting)

This item sets the frames and field in which noise is generated.

Broadcast :Multiple-address bit. 0 indicates multiple-address communication and 1 indicates individual communication.
Master :Master address
Slave :Slave address
Control :Control bit
Size :Size
Data :Data

Error Point :Specifies the field in which noise is generated.
Slave Address :Generates a timing error using the slave address.
Control Bit :Generates a timing error using the control bit.
Size :Generates a timing error using the size.
Data byte1～9 :Generates a timing error using byte positions in which data is specified.

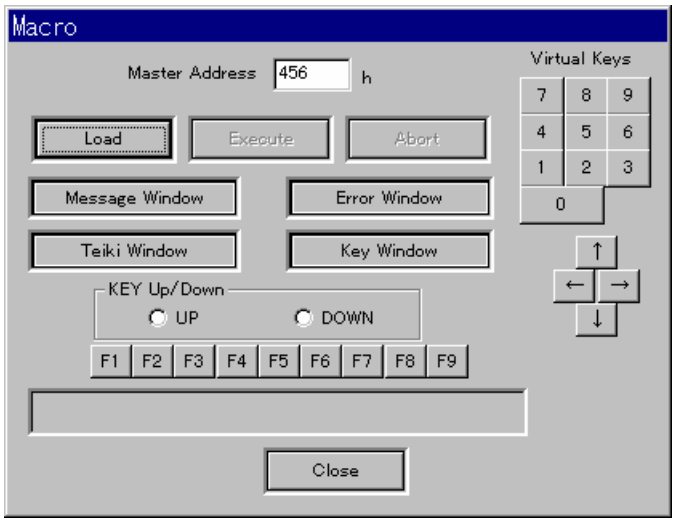


In the above setting, a timing error occurs in the fourth byte of the frame data.
Multiple-address communication.
Master address = 123h
Slave address = FFFh
Control bit = Fh
Size =06h
1st data byte = 00h
2nd data byte = 00h
3rd data byte = 00h

※ Data matching is not checked for the error point (in this example, the fourth byte).

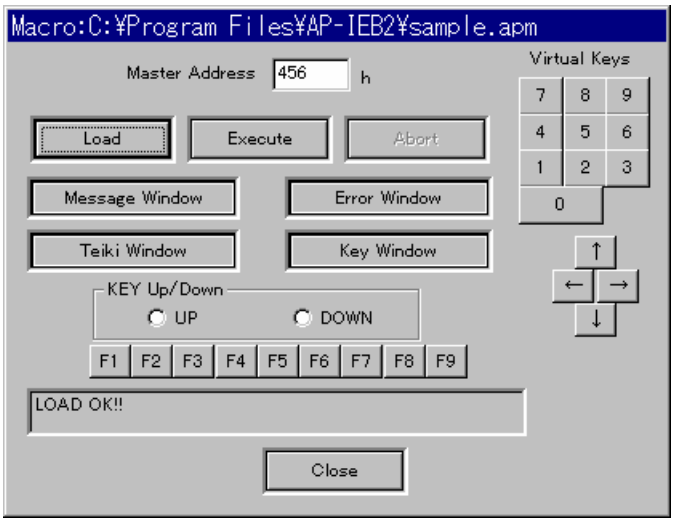
5.7. Macros (executing the macro language)

When a macro is executed, the following window appears.



Master Address　　Specified the master address of the device in which the macro is executed.

［Load］ button　　Specifies the macro file to be executed.
When the macro file is read, the file name is displayed in the title bar as shown below and the macro is available for execution



［Execute］ button　　Executes a loaded macro.
［Abort］ button　　Aborts (force-quits) a macro being executed.
［Message Window］ button　　Displays a message window. ECHO characters and other characters are displayed.
［Error Window］ button　　Displays an error message window.
Error codes and error lines are displayed in this window when macros are executed.
［Teiki Window］ button　　Displays a “Teiki” message window for scheduled instructions
Displays only the first command executed during scheduled communication processing.

［Key Window］ button	Displays a key message window. When macro-defined key input occurs, the first instruction for execution is displayed.
UP/DOWN box	This box sets events executed when buttons 0-9, arrows or F1-F9 displayed in the window are clicked. When DOWN is set, each button click is processed as if the key had been pressed. When UP is set, each button click is processed as if the key had been released.
Virtual Keys	Clicking these buttons performs branching using the strings KEYGOTO and KEYGOSUB. AP-IEB2 does not respond to actual keyboard input of the strings KEYGOTO and KEYGOSUB.
[Close]	Quits the macro and closes the window.

Note:

- Frame Start in Trace is not available during macro execution.
- Macros can be executed even when monitoring is not being executed.

6. Function menu

Click Function on the menu bar of the main menu.
Submenu items appear.
Click the desired item as described below.

Name of submenu item

① Trigger Data Load
Data Save

This item loads and saves trigger data.
Data set the last time the program was started is automatically saved as IEBus. trd in the directory in which AP-IEB2 is installed and is loaded automatically when AP-IEB2 is started.

② Frame Data Load
Data Save

This item loads and saves frame data.
Data set the last time the program was started is automatically saved as IEBus. frd in the directory in which AP-IEB2 is installed and is loaded automatically when AP-IEB2 is started.

③ Trace Data Load
Data Save

This item loads and saves trace data.
Trace data can be loaded and saved in the following formats.

ie 2 : New format. This is the format that should be normally used for saving trace data.
ie1 : Previous format, identical to AP-PAB.

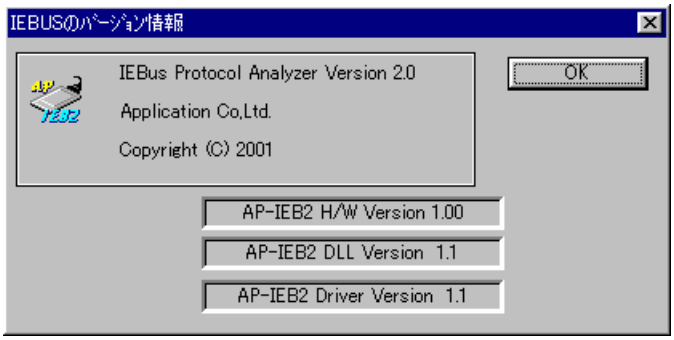
Trace data can be saved only in the following format.

txt : Data are arranged as displayed in the window. This format is convenient for printing.

④ Macro Open Macro Window

7. Help Menu

The Help window is displayed when Help is selected from the menu bar. This can be used to check your version of AP-IEB2.



8. Macro Language Specifications

This section describes the AP-Macro macro language. AP-Macro enables one command to be written on a single line, which AP-IEB2 then interprets and executes.

8.1. Comments

In AP-Macro, all character after '#' are comments. If '#' appears at the beginning of a line, the entire line is handled as a comment. If '#' appears elsewhere in the line, only the part of the line after the '#' is a comment. Comments are ignored during execution.

Example:

```
# Frame transmission
SEND frame0
RECV frame1  # A reply is received in response to the sent frame
```

8.2. Formats and statement of variables

Variables can be used in AP-Macro. Formats include **IDATA**, **INT**, **IFRAME** and **IFMASK**.
The first character of a variable must be a lower-case alphabetic character. Only alphabetic characters and underlines can be used in variables, to a maximum of 255 characters.
In AP-Macro, statements of variables used are contained in macro description files. These are followed by procedure statements (see below), then by the process description. These statements must appear in the macro in this order.

＜**IDATA** format＞

IDATA format is an array with a specific byte size of 255Byte. It is used to define the IEBus frame data array. Each entry is an 8bit value containing no symbols. Non-initialized values are set to zero.

Example:

```
IDATA a={0x00,0x12,0x13}
a[10]=0x55
```

＜**INT** format＞

INT format is used to define integers. Internally this format has a size of 32bit (4Byte) with symbols. It can handle values in the range from –2G (giga) to +2G. **INT** format variables can be substituted for each entry in the **IDATA** format and each member (adr, etc.) in the **IFRAME** format. In this case only the final bit of the 3.2bit of the **INT** format is substituted as a non-symbol number.

Example:

```
IDATA a
INT y=1
a[2]=y
```

INT format alone can be handled as both a one-dimensional and a two-dimensional array.

Example:

```
INT y[2]
y[0]=50
y[1]=30

INT x[2][3]
x[0][2]=50
x[1][2]=30
```

＜**IFRAME** format＞

IFRAME format defines the IEBus frame. Internally it is a structure consisting of the following members. Access to each member is performed in the same way as in C language. In all members, uninitialized values are set to zero.

```
IFRAME {
  INT bit          Multi-address bit
  INT adr          Slave address
  INT size         String length
  IDATA data       Frame data array
}
```

Example:

```
IFRAME frame0
IDATA a
frame0.data[3]=5
frame0.data=a
IFRAME frame1={1,0x123,5,0x00,0x01,0x02}
```

Supplementary note:

In the above, 1 is a multi-address bit. 0x123 is a slave address. 5 is string length. Subsequent items 0x00, 0x01… are data. They are saved internally in a fixed 255Byte area in the same way as the DETA format. Uninitialized values are set to zero.

In multi-address bits, 0 indicates multi-address and 1 indicates and individual address.

＜**IFMASK** format＞

IFMASK format defines the IEBus frame mask. Using the **CHECK** command described below, this format is used to perform a comparative check of the masked portion of received frames with a specified value. Because the internal structure is the same as the **IFRAME** format, access and other methods are identical.

Example:

```
IFRAME checkframe={1,0x123,5,0x00,0x01,0x02}
IFRAME recvframe
IFMASK fmask={0,0xffff,0x0,0xff,0xff,0xff}
RECV recvframe
CHECK checkframe recvframe fmask
```

Supplementary note:

In the IFMASK statement shown above, the multi-address bit is unmasked (not subject to checking), the address is masked (subject to checking), the string length is unmasked (not subject to checking) and the first 3bytes of the data is masked (subject to checking).

The **CHECK** command performs an AND comparison of the frame received using the RECV command and the frame stated as **checkframe**.

8.3. Constants

Constants can be used in AP-Macro. Constant values have the same specifications as in C language. Numbers beginning in 0x are hexadecimals, numbers beginning in 0 are octadecimals and numbers with no prefix are decimals.
Constants are used as initial value for variables, substitute values and shift values.

8.4. Labels

Labels can be used in AP-Macro. Labels are used as destinations for commands such as the **GOTO** command and **IF-THEN** command (see below) to indicate macro files. The scope of labels is within procedures only (see below); labels cannot be linked from the main processing section to the inside of a procedure.

例:
label0:

As shown above, the line ends in ":" and begins with a label name in lower-case alphabetic characters. The line that describes the label cannot describe other commands.

8.5. Procedure

AP-Macro can describe a set of processes called procedures. Procedures can be registered as scheduled processing procedures and key processing procedures (see below) and are called as subroutines for the GOSUB command.

Example:
PROC proc0
・・・Description of processing
ENDPROC

A procedure always begin with **PROC** and ends in **ENDPROC**. **GOTO** can only link to a label in a **PROC** if it is in the same PROC (the same applies to **IF** labels). Setting and canceling of scheduled transmissions and key input cannot be performed from within a **PROC**.

8.6. Operations

Variable substitutions and operations can be described in AP-Macro. The types of operations that can be described on a single line as a formula are as follows.

a[3] += 5	Addition of constants
a[3] -= 5	Subtraction of constants
a[3] |= 5	Logical “OR” of constants
a[3] &= 5	Logical “AND” constants
a[3] /= 5	Division of constants
a[3] *= 5	Accumulation of constants
a[3] <<= 5	Left-bit shift
a[3] >>= 5	Right-bit shift
a[3] = 5	Substitution of constants
a[3] += b	Addition of variables
a[3] -= b	Subtraction of variables
a[3] |= b	Logical “OR” variables
a[3] &= b	Logical “AND” variables
a[3] /= b	Division of variables
a[3] *= b	Accumulation of variables
a[3] <<= b	Left-bit shift
a[3] >>= b	Right-bit shift
a[3] = b	Substitution of variables

The variable formats on the left side and right side must be the same.

The left side, operator and right side must be separated by a space or TAB. Operators such as ‘+’ must not be separated by a space.

The operations logical AND, logical OR and bit shift can be handled as variables with symbols in the **INT** format or as values without symbols.

8.7. Flow control commands

Flow control commands in AP-Macro are as follows.

＜SWITCH～CASE Command＞

A variable name must be specified as a parameter for the **SWITCH** command. The variable format can specify each entry in **INT** format and **IDATA** format and every member in **IFRAME** format and **IFMASK** format.
A nested structure, in which a **SWITCH-ENDSWITCH** command is included in another **SWITCH-ENDSWITCH** command, is possible.

Example:

```
IDATA    abc
SWITCH abc[3]
 CASE 0
    ・・・Description of processing
 ENDCASE
 CASE 0x11
    ・・・Description of processing
 ENDCASE
 CASE 0x22
    ・・・Description of processing
 ENDCASE
 CASE DEFAULT
    ・・・Description of processing
 ENDCASE
ENDSWITCH
```

＜IF～THEN command＞

This command jumps to a label described after THEN according to the result of evaluation of a formula described after IF. In this case a variable with symbols is handled as a variable with symbols and a variable without symbols is handled as a variable without a symbols.

Example:

```
IF a != 0 THEN label0
IF a == 0 THEN label0
IF a & 1 THEN label0
IF a < 1 THEN label0
IF a > 1 THEN label0
IF a <= 1 THEN label0
IF a >= 1 THEN label0
```

＜WHILE command＞

While the result of an evaluation of a format described after WHILE is true, execution continues until ENDWHILE. A nested structure, in which a WHILE-ENDWHILE command is included in another WHILE-ENDWHILE command, is possible.

Example:

```
WHILE a!=0
   ・・・Description of processing
ENDWHILE
```

＜**GOTO** command＞

Jump to the label described after **GOTO**.

Example:

```
GOTO label0
```

＜**EXIT** command＞

Aborts macro processing. This command exits from a macro without specifying a parameter. This command cannot be used within a procedure.

Example:

```
EXIT
```

8.8. Other commands

In addition to the CVS-format file command described below, AP-Macro provides the following commands.

＜**WAIT** command＞

Causes the program to sleep for the specified number of milliseconds.

Example:

```
WAIT 100
```

＜**SEND** command＞

Send the specified number of frames

Example:

```
SEND frame0
```

＜**RECV** command＞

Receives frames according to the specified variable and specifies a timeout value in milliseconds. The timeout can be omitted, in which case no timeout occurs and the system waits until the frames are received.

Example:

```
RECV frame0 100
```

＜**CHECK** command＞

Checks only part 1, using a pattern specified in the variable of the IFMASK format. If the compared result is equal, 0 is set to the system definition variable **errno** (see below). If it is not equal, 1 is set.

Example:

```
IFRAME checkframe={1,0x123,5,0x00,0x01,0x02}
IFRAME recvframe
IFMASK fmask={0x0,0xffff,0x0,0xff,0xff,0xff}
RECV recvframe
CHECK checkframe recvframe fmask
IF errno != 0 THEN label_error
ECHO check OK
    ・・・Description of processing
label_error:
ECHO check error
```

Supplementary note:

In the IFMASK statement shown above, the multi-address bit is unmasked (not subject to checking), the address is masked (subject to checking), the string length is unmasked (not subject to checking) and the first 3 bytes of the data is masked (subject to checking).

The CHECK command performs an AND comparison of the frame received using the **RECV** command and the frame stated as **checkframe**.

＜**KEYGOTO** command＞

Defines a destination label to which to jump when a specified key is pressed of released. The number keys 1 to 9, function keys F1 to F9, up/down/left/right arrow keys and alphabetic keys A to Z can be specified (case-insensitive).

Example:

KEYGOTO 0 label0 (u)

The 'u' after the label can be omitted. When specified, it defines execution when the key is released.

＜**KEYGOSUB** command＞

Defines a key processing procedure that is executed when a specified key is pressed or released. The number key 1 to 9, function keys F1 to F9, up/down/ left/ right arrow keys and alphabetic keys A to Z can be specified (case-insentive).

Example:

KEYGOSUB 0 proc0 (u)

The 'u' after procedure name can be omitted. When specified, it defines execution when the key released.

＜**BEEP** command＞

Causes a system-defined beep to sound. Sounds are each of the sounds allocated in the parameters minfo (message information), syserr (system error),mques (question), OK (general warning noise), found in Control Panel – Sounds. Sounds are not generated when the sound is muted. The parameter pc specifies a beep in the PC's built-in speaker. All of these sounds can be specified.

Example:

BEEP pc/minfo/mwarn/syserr/mques/ok

＜**TEIKI** command＞

Registers a scheduled processing procedure. Number 0 to 9 can be registered. The final parameter is an interval defined in milliseconds.

Example:

TEIKI 0 proc0 100

＜**TSTOP** command＞

Discards registration of the scheduled processing procedure.

Example:

TSTOP 0

＜**ECHO** command＞

Indicates a specified string in the debug string display window.

Example:

```
ECHO This is a debug program
```

＜**GOSUB** command＞

Calls a procedure. The procedures called are stored in the internal stack, so further procedures can be called from within procedures.

Example:

```
GOSUB proc0
```

＜**XOR** command＞

Performs an exclusive OR operation on the variable specified on the left and the variable specified on the left and the variable or constant specified on the right, then substitutes the result on the left side. The specified variable or constant is handled as a value without symbols. If the specified variable is an array variable, only one element is specified.

Example:

```
INT a=0xffffffff
INT b=0xCCCCCCCC
INT c[2][3]
XOR a 0x30303030
XOR a b
c[1][0]=0xdddddddd
XOR c[1][0] b
```

＜**INV** command＞

Substitutes the result of a bit inversion on a specified variable. The specified variable or constant is handled as a value without symbols. If the specified variable is an array variable, only one element is specified.

Example:

```
INT a=0xffffffff
INT c[2][3]
INV a
c[1][0]=0xdddddddd
INV c[1][0]
```

＜**INCLUDE** command＞

Includes a file and deploys it at that location. Any file name or extension can be used. The full path is specified in the area enclosed by “”. If the path is omitted the file is included in the current directory.

Example:

```
INCLUDE “file.inc”
INCLUDE “c:¥user¥file.inc”
```

＜**DEFINE** command＞

Defines a replacement string for a variable or constant. As with variables, the replacement string must begin with a lower-case alphabetic character. The replacement string must not overlap with the name of a variable, label, PROC or command. Up to 256 replacement strings can be defined.
Defining only the members of a variable is also possible. Character strings enclose in “” cannot be defined.

Example:
DEFINE abcd 1
DEFINE aa55 frame.data[1]

＜**TIMEGOSUB** command＞

Defines a procedure to be executed after a specified interval (specified in milliseconds). Numbers can be registered from 0 to 9. The timer is executed once only and is used for timeout processing. For cyclical timer processing, use the **TEIKI** command.

Example:
TIMEGOSUB 0 proc0 800

＜**TIMEGOTO** command＞

Defines a label to change the position of execution after a specified interval (specified in milliseconds). Numbers can be registered from 0 to 9. The timer is executed once only and is used for timeout processing. For cyclical timer processing, use the **TEIKI** command. As in other cases, label scope is restricted to within the same PROC and within main processing.

Example:
TIMEGOTO 0 label0 800

＜**TIMESTOP** command＞

Discards registration of procedures and label branching defined in the **TIMEGOSUB** command and **TIMEGOTO** command. Numbers from 0 to 9 can be used.

Example:
TIMESTOP 0

8.9. CSV format file commands

In AP-Macro, CSV-format files can be used. CSV-format files are files consisting of a single line in a single text file, with each field separated by a comma. Commands for using CSV-format files are as follows.

＜**COPENR** command＞

Opens a CSV-format file in read mode. Any file name or extension can be used. The full path is specified in the area enclosed by “”. If the path is omitted the file is included in the current directory. Reading of all subsequent file is conducted for this file.

Example:
COPENR “c:¥user¥frame.csv”

＜**COPENW** command＞

Opens a CSV-format file in write mode. Any file name or extension can be used. The full path is specified in the area enclosed by “”. If the path is omitted the file is included in the current directory. Writing of all subsequent files is conducted for this file.

Example:

COPENW “c:¥user¥frame.csv”

＜**CCLOSE** command＞

Closes a file opened in COPENR or COPENW. Separated by ‘R’ or ‘W’.

Example:

CCLOSE R

CCLOSE W

＜**CLOAD** command＞

Reads one record (one line) to a variable specified from a file opened in read mode. Array members such as **“data0[3]”** and “**frame.bit”** can be specified, and single members of an array can be specified. Remaining field data in a record (data that does not fit in the variable) is discarded.

In one-dimensional and two-dimensional variables in **IDATA** format or **INT** format, this command can specify an entire array or two-dimensional array instead of a signal element.

Example:

INT int0

INT int1[2]

INT int2[2][3]

CLOAD frame0

CLOAD frame0.bit

CLOAD data0

CLOAD data0[2]

CLOAD int0

CLOAD int1

CLOAD int1[1]

CLOAD int2

CLOAD int2[1]

CLOAD int2[1][1]

＜**CSAVE** command＞

Writes one record (one line) from the contents of a variable specified in a file opened in write mode. Array element such as **"data0[3]"** and "**frame.bit"** can be specified, and single members of an array can be specified.

In one-dimensional and two-dimensional variables in **IDATA** format or **INT** format, this command can specify an entire array or two-dimensional array instead of a signal element.

Example:

```
INT int0
INT int1[2]
INT int2[2][3]
CSAVE frame0
CSAVE frame0.bit
CSAVE data0
CSAVE data0[2]
CSAVE int0
CSAVE int1
CSAVE int1[1]
CSAVE int2
CSAVE int2[1]
CSAVE int2[1][1]
```

8.10. System Definition Variables

AP-Macro includes system definition integer variables such as **errno**. These variables store error values generated during execution of commands such as the **CHECK** command. These can be used in combination with commands such as the **CHECK** command and **IF-THEN** command.

8.11. Sample Macro

The following is a sample of a macro file that can be described in the AP-Macro language. The line numbers to the left of the text of the macro file are for the reader's convenience only and are not included in the actual macro file.

Macro file

```
 1 : #
 2 : # Sample Macro
 3 : #
 4 :
 5 : # Variable statement
 6 : IFRAME  fSend0 = {1, 0x123, 5, 0x00, 0x11, 0x22, 0x33, 0x44}
 7 : IFRAME  fSend1 = {1, 0x123, 3, 0x56, 0x78, 0x9a}
 8 : IFRAME  fSend2 = {1, 0x123, 1, 0xff}
 9 : IFRAME  fSend3 = {1, 0x123, 1, 0x55}
10 : IFRAME  fSend4 = {1, 0x123, 1, 0xee}
11 : IFRAME  fCmp =   {0,      0x123,   0,      0x00, 0x42}
12 : IFMASK  fMask = {0x00, 0xFFFF, 0x00, 0x00, 0xFF, 0x00, 0x00, 0x00}
13 : IFRAME  fRecv
14 : INT   cnt
15 :
16 : # Scheduled transmission procedure
17 : PROC     teikiProc0
18 :   cnt = 3
19 :   WHILE cnt!=0        # Repeated 3 times
20 :   SEND   fSend1
21 :   cnt -= 1
22 :   ENDWHILE
23 : ENDPROC
24 :
25 : # 5 Procedure when the 5 key is prssed.
26 : PROC     key5Proc
27 :   SEND   fSend2
28 : ENDPROC
29 :
30 : # Processing
31 :
32 : KEYGOSUB 5 key5Proc          # 5 Defines processing when the 5 key is pressed
33 :
34 : label0:
35 : RECV     fRecv          # Waiting to receive frame that matches conditions.
36 : IF errno!=0 THEN error
37 : CHECK   fRecv fCmp fMask
38 : IF errno==0 THEN label1
39 : GOTO label0
40 :
41 : label1:
42 : WAIT 500
43 : SEND fSend2
```

```
44 : IF errno!=0 THEN error
45 : label2:
46 : RECV fRecv              # Processing is changed by the first byte received.
47 : IF errno!=0 THEN error
48 : SWITCH fRecv.data[0]
49 :   CASE 0x11
50 :       TEIKI 0 teikiProc0 5000
51 :       GOTO label3
52 :   ENDCASE
53 :   CASE 0x22
54 :       SEND fSend3
55 :       IF errno!=0 THEN error
56 :   ENDCASE
57 :   CASE DEFAULT
58 :       SEND fSend3
59 :       IF errno!=0 THEN error
60 :   ENDCASE
61 : ENDSWITCH
62 : GOTO label2
63 :
64 : label3:
65 : RECV     fRecv          # First byte of data received waits for 0x88
66 : IF errno!=0 THEN error
67 : IF fRecv.data[0]==0x88 success
68 : SEND fSend3
69 : IF errno!=0 THEN error
70 : goto label3
71 :
72 : error:
73 :   EXIT
74 : success:
75 :   EXIT
```

Line-by-Line description

Lines 1 to 5:

These are comment lines and spaces. They are ignored during processing.

Lines 6 to 14:

States the variables used in this macro. In AP-Macro, a statement of all variables must precede a description of the processing.

Initialization data that is omitted is set to an initial value of zero. The initialization data for the **IFRAME** format is set in the data section, in order from left to right, multi-address bit, address and string length. Omitted data is set to an initial value of zero.

In AP-Macro, all variables are global variables. The concept of "scope of variables" does not apply. This is also true within procedures. All variables can be linked to in common, with no distinctions.

Lines 17 to 23:

This is a procedure used later, for scheduled processing. Here the IFRAME format variable **fSend1** is sent three times.

Lines 25 to 28:

This is a procedure use later, for key processing. Here the **IFRAME** format variable **fSend1** is sent.

Line 30 forward:

Main processing is executed starting here.

Line32:

Defines key processing. After this point, if the 5 key is pressed, **key5Proc** is started.

Line 33:

Defines a label.

Lines 35 and 36:

Receives the **fRecv** variable. The next line checks whether an error occurs in the RECV command in the previous line. If an error is found, processing jumps to the label 'error' in line

Line 37 to 39:

Performs an AND operation with the **fMask** variable on the frame received in the **fRecv** variable and checks by comparing it with the contents of the **fCmp** variable. The next line checks whether it is equal to the result of the **CHECK** command on the previous line. If it is equal, processing jumps to label 1 on line 41. Otherwise, the **GOTO** command on the next line returns processing to **label0** on line 34 and receiving check is repeated.

Lines 42 to 44:

Waits 500msec due to the **WAIT** command, then sends the contents of the **fSend2** variable. If a transmission error occurs, processing jumps to label '**error**' in line72.

Lines 45 to 62:

Here the **SWITCH-CASE** command is used to separate processing of the data in the received frames according to the first byte. If the first byte of data is 0x11, the system is set to start the **teikiProc** procedure every five seconds using the **TEIKI** command and proceed to '**label3**' on line 64. If the first byte of data is 0x22, after the contents of the **fSend3** variable are sent, processing is returned to '**label2**' using **GOTO** and the system waits to receive data again. If the first byte of data is neither 0x11 nor 0x22, **CASE DEFAULT** treats the data as if its first byte were 0x22.

Lines 64 to 70:

Here the **IF-THEN** command is used to separate processing of the data in the received frames according to the first byte. If the first byte of data is 0x88, processing jumps to line 74 and macro processing is ended. Otherwise processing returns to '**label3**' on line 64 after the contents of the fSend3 variable are sent.

Line 72:

If reception or transmission errors occur in the execution of this macro, processing jumps to the label '**error**'.

Line 74:

If all processing in the execution of this macro is completed correctly, processing jumps to the label '**success'.**