APPENDIX A. KIT-NB85E-TP INTERNAL COMMANDS

This appendix describes the KIT-NB85E-TP internal commands. These commands can be used as through commands in the debugger. For an explanation of using through commands, refer to the manual provided with the debugger.

With PARTNER/Win

```
>&          << Enter through command mode.
>#ENV       << Enter an internal command.
>&          << Exit from through command mode.
```

With GHS-Multi

The through commands can be directly input in the target window after RTESERV has been connected.

Commands

Note These commands can be used only if the debugger does not provide equivalent functions. If these commands are issued when the debugger provides equivalent functions, a contention may occur between KIT-NB85E-TP and the debugger, causing either device to malfunction.
Each command of eva/eve/evt/seq is a command which corresponded more than by V5.10.xx of rte4win32.

Command syntax

The basic syntax for the KIT-NB85E-TP internal commands is described below:

command-name parameter(s)

* In parameter syntax, a parameter enclosed in brackets ([ ]) is omissible. A horizontal line (|) indicates that one of the parameters delimited by it must be selected.

A command name must be an alphabetic character string, and be separated from its parameter(s) by a space or tab. A parameter must be an alphabetic character string or hexadecimal number, and be delimited by a space or tab. (A hexadecimal number cannot contain operators.)

abp, abp1, and abp2 commands

[Format]

```
abp [or|and|seq]
abp {1|2}     [ADDR [AMASK]] [data DATA [DMASK]] [asid ASID|noasid] [aeq|aneq] [deq|dneq]
              [exec|read|write|accs] [byte|hword|word|nosize]
abp {1|2}     /del
```

[Parameters]

abp [or|and|seq]:	Specifies a condition for combination of abp1 and abp2.
or:	Break occurs if either abp1 or abp2 occurs.
and:	Break occurs if both abp1 and abp2 occur at the same time. A mask condition is used.
seq:	Break occurs if abp2 occurs after abp1.
abp {1|2}:	Input before the condition of abp1 or abp2 is specified.
ADDR [AMASK]:	Specifies an address condition.
ADDR:	Specifies addresses in hexadecimal number.
AMASK:	Specifies the mask data of an address in hexadecimal. Bits that are 1 will not be compared.
data DATA [DMASK]:	Specifies a data condition.
DATA:	Specifies data in hexadecimal.
DMASK:	Specifies the mask data of data in hexadecimal. Bits that are 1 will not be compared.
asid ASID|noasid:	For future expansion. Use noasid.
aeq|aneq:	Specifies an address comparison condition.
aeq:	Compares address for equality.
aneq:	Compares address for non-equality.
deqldneq:	Specifies a data comparison condition.
deq:	Compares data for equality.
dneq:	Compares data for non-equality.
exec|read|write|accs:	Specifies a cycle condition.
exec:	Specifies an executable address. A data condition is ignored.
read:	Specifies a read cycle.
write:	Specifies a write cycle.
accs:	Specifies a read or write cycle.
byte|hword|word|nosize:	Specifies access size.
byte:	Specifies byte access (8 bits).
hword:	Specifies half-word access (16 bits).
word:	Specifies word access (32 bits).
nosize:	Specifies invalidity.
abp{1|2}/del:	Clears a condition.
/del:	Specifies deletion of a condition.

[Function]

These commands set or delete access break points.

Up to two access break points can be set.

They can specify execution addresses.

[Examples]

abp or

abp1 or abp2 is specified.

abp1 1000 aeq exec

A breakpoint for execution of address 1000h is set.

abp2 1000 data 5555 0 aeq deq read hword

Break occurs when 5555h is read in hword from address 1000h.

abp1 /del

The condition set by abp1 is deleted.

env and ememstat commands

[Format]

```
env    [[!]auto] [[!][verify]] [[!]reset] [[!]stopz] [[!]hldrq]
       [jtag{25|12|5|2|1|500|250|100}] [[!]nmi0] [[!]nmi1] [[!]nmi2]
       [rtrcb{0|25|50|75}] [nrtrcb{12|25|37|50}] [64m|256m]
       [romless|single0|single1] [d0|d1|d2|dauto] [i0|i1|i2|iauto]
```

[Parameters]

[!]auto: If a break point is set during execution, the break point causes a temporary break. Choose [auto] to automatically perform the subsequent execution. Choose [!auto] to suppress it.

[!]verify: Specifies whether the verification after writing memory is set. Enter ! if it is not to be set.

> **Remark** The CPU also accesses an area that emulates ROM (jread or equivalent). Therefore, this command is also useful for testing the area during downloading. Note, however, that the processing speed slows down.

[!]reset:	Specifies whether the RESET pin is to be masked. Enter ! if it is not to be masked.
[!]stopz:	Specifies whether the STOPZ pin is to be masked. Enter ! if it is not to be masked.
[!]hldrq:	Specifies whether the VAREQ pin is to be masked. Enter ! if it is not to be masked.
[!]nmi{0|1|2}:	Specifies whether pins NMI0 to NMI2 are to be masked. Enter ! if they are not to be masked.

jtag{25|12|5|2|1|500|250|100}: Specifies the JTAG clock for N-Wire. Each number corresponds to the following JTAG clock.
[25 MHz|12.5 MHz|5 MHz|2 MHz|1 MHz|500 kHz|250 kHz|100 kHz]

> **Remark** Usually, use 25 MHz or 12.5 MHz. If the frequency lower than 1 MHz is specified, the debugger might be slowed down in operation speed or might malfunction. And for RTE-100-TP, the parameters other than jtag-25 or jtag12 are invalid.
> The initial value depends on the version of rte4win32.
> V5.02 or earlier : It is always 12.5 MHz.
> V5.03 or later : It is automatically set to the highest frequency at which the machine operates.

rtrcb {0|25|50|75}: Specifies the occupied capacity of the buffer when execution returns from overflow during real-time trace. Usually, use the initial value of this parameter.

nrtrcb {12|25|37|50}: Specifies the occupied capacity of the buffer when a request to stop the pipeline is made in complete trace mode. Usually, use the initial value of this parameter.

64m|256m: Specifies an address mode of the CPU.

64m:	Specifies the 64M mode.
256m:	Specifies the 256M mode.

romless|single0|single1: Specifies an operation mode of the CPU.

single0m:	Specifies the single mode 0 (internal ROM from address 0).
single1:	Specifies the single mode 1 (internal ROM from address 100000h).
Romless:	Specifies the ROM-less mode.

d0|d1|d2|dauto: Specifies data cache.

d0:	Specifies no data cache.
d1:	Specifies the cache of direct map.
d2:	Specifies the 2-WAY cache.
dauto:	Specified in the case of NB85E-TEG for automatic setting.

i0|i1|i2|iauto:	Specifies instruction cache.
i0:	Specifies no instruction cache.
i1:	Specifies the cache of direct map.
i2:	Specifies the 2-WAY cache.
iauto:	Specified in the case of NB85E-TEG for automatic setting.

Remark To specify dauto or iauto is limited for the evaluation board using NB85E-TEG chip. Usually specify the cache mode that is actually implemented to the CPU.

[Function]

The env command sets the emulation environment and displays the DCU status.

Enter only those parameters that need to be changed. Parameters may be entered in any order.

If the same parameter is entered twice, only the last entry is valid.

The ememstat command displays the mounting status of the E.MEM board when RTE-2000-TP is used.

Display examples are shown below:

With RTE-1000-TP

```
Probe:
 Unit          : RTE-1000-TP        << Displays the main unit connected.
 Rom Probe : Extend Type          << Displays the ROM probe type connected.
 Emem Size : 32Mbyte               << Displays the size of emulation memory implemented.
CPU Settings:
 Auto Run       = ON (auto)
 JTAGCLOCK  = 12.5MHz (jtag12)
 Verify          = verify off (!verify)
 CPU Mode      = romless (romless)        << Depends on rte4win32 configuration.
 Space           = 64M Byte Mode (64m)   << Depends on rte4win32 configuration.
Signals Mask:
 NMI0            = NO MASK (!nmi0)
 NMI1            = NO MASK (!nmi1)
 NMI2            = NO MASK (!nmi2)
 RESET          = NO MASK (!reset)
 HLDRQ          = NO MASK (!hldrq)
 STOPZ          = NO MASK (!stopz)
Trace Buffer Usage Settings:
 Realtime       <=  0% (rtrcb0)
 None Realtime>= 12% (nrtrcb12)
Trace UNIT:
 Cotrol Unit     = Enable
 Event Unit      = Enable
  Execute    Event Number  = 8
  Access     Event Number  = 4
  Sequence Event Number  = 1
  Sequence Counter Bit      = 12
Cache Mode:
 Data            = 2Way (d2)
 Instruction    = Auto Detect (iauto)
```

With RTE-2000-TP

```
Probe:
 Unit          : RTE-2000-TP                    << Displays the main unit connected.
 Rom Probe : (use ememstat command)
 Emem Size : (use ememstat command)
CPU Settings:
 Auto Run       = ON (auto)
 JTAGCLOCK  = 25MHz (jtag25)
 Verify          = verify off (!verify)
 CPU Mode      = single0 (single0)              << Depends on rte4win32 configuration.
 Space           = 64M Byte Mode (64m)        << Depends on rte4win32 configuration.
Signals Mask:
 NMI0            = NO MASK (!nmi0)
 NMI1            = NO MASK (!nmi1)
 NMI2            = NO MASK (!nmi2)
 RESET          = NO MASK (!reset)
 HLDRQ          = NO MASK (!hldrq)
 STOPZ          = NO MASK (!stopz)
Trace Buffer Usage Settings:
 Realtime       <=  0% (rtrcb0)
 None Realtime>= 12% (nrtrcb12)
Trace UNIT:
 Cotrol Unit     = Enable
 Event Unit       = Enable
  Execute    Event Number = 8
  Access     Event Number = 4
  Sequence Event Number = 1
  Sequence Counter Bit  = 12
Cache Mode:
 Data             = Auto Detect (dauto)
 Instruction      = Auto Detect (iauto)

rte3>emustat
Command Not Found.
rte3>ememstat
 Board_num  EMEM_Size  ROM_Probe
 ===================================
      ROM1       8Mbyte        Extend Type 2K
```

[Examples]

env reset !nmi0 verify

RESET is masked while NMI0 is not masked. Sets the Verify function to ON.

eva command

[Format]

```
eva  {1..8} [ADDR] [data DATA [MASK]] [asid ASID|noasid] [eq|lt|gt|neq|lte|gte|ign]
     [deq|dneq] [read|write|accs] [byte|hword|word|nosize] [/del]
```

[Parameters]

eva {1..8}:	Specifies an access event channel (Nx85ET can use only 1-4 ch.).
ADDR:	Specifies the address in hexadecimal.
data DATA [MASK]:	Specifies a data condition.
DATA:	Specifies data in hexadecimal.
MASK:	Specifies mask data for the data in hexadecimal. Bits that are 1 will not be compared.
asid ASID|noasid:	For future expansion. Use noasid.
eq|lt|gt|neq|lte|gte|ign:	
eq:	Specifies that the condition is satisfied when the event address is equal to the address specified for ADDR.
lt:	Specifies that the condition is satisfied when the event address is smaller than the address specified for ADDR.
gt:	Specifies that the condition is satisfied when the event address is greater than the address specified for ADDR.
neq:	Specifies that the condition is satisfied when the event address is not equal to the address specified for ADDR.
lte:	Specifies that the condition is satisfied when the event address is smaller than or equal to the address specified for ADDR.
gte:	Specifies that the condition is satisfied when the event address is greater than or equal to the address specified for ADDR.
ign:	Specifies that ADDR is not used as a comparison condition.
deq|dneq:	Specifies a data comparison condition.
deq:	Compares data for equality.
dneq:	Compares data for non-equality.
read|write|accs:	Specifies a cycle condition.
read:	Specifies a read cycle.
write:	Specifies a write cycle.
accs:	Specifies a read or write cycle.
byte|hword|word|nosize:	Specifies access size.
byte:	Specifies byte access (8 bits).
hword:	Specifies half-word access (16 bits).
word:	Specifies word access (32 bits).
nosize:	Specifies invalidity.
eva {1..6} /del:	Clears a condition.
/del:	Specifies deletion of a condition.

[Function]

The eva command sets an access event. The specified event can be combined with a condition using the evt command to be used as a break or trace condition.

[Examples]

eva 1 ffff000 data 55 00 byte read

A cycle for reading 0x55 starting at address 0xffff000 is set for eva ch1 with using the default values for other parameters.

eva 1 /del

The condition of eva ch1 is cleared.

eve command

[Format]

eve {1..16} [ADDR] [asid ASID|noasid] [eq|lt|gt|neq|lte|gte|ign] [/del]

[Parameters]

eve {1..16}:	Specifies an execution event channel.(Nx85ET can use only 1-8 ch.)
ADDR:	Specifies the address in hexadecimal.
asid ASID|noasid:	For future expansion. Use noasid.
eq|lt|gt|neq|lte|gte|ign:	
eq:	Specifies that the condition is satisfied when the event address is equal to the address specified for ADDR.
lt:	Specifies that the condition is satisfied when the event address is smaller than the address specified for ADDR.
gt:	Specifies that the condition is satisfied when the event address is greater than the address specified for ADDR.
neq:	Specifies that the condition is satisfied when the event address is not equal to the address specified for ADDR.
lte:	Specifies that the condition is satisfied when the event address is smaller than or equal to the address specified for ADDR.
gte:	Specifies that the condition is satisfied when the event address is greater than or equal to the address specified for ADDR.
ign:	Specifies that ADDR is not used as a comparison condition.
eve {1..16} /del:	Clears a condition.
/del:	Specifies deletion of a condition.

[Function]

The eve command sets an execution event. The specified event can be combined with a condition using the evt command to be used as a break or trace condition.

[Examples]

eve 1 1000

Execution of the instruction at address 0x1000 is set for eve ch1 using the default values for other parameters.

eve 1 /del

The condition of eve ch1 is cleared.

evt command

[Format]

```
evt   {brk|seqclr|seq1|seq2|seq3|seq4|trcs1trcs2|trcr|trg|match}
      evep{[1][2][3]..[g]} ever{[1][3][5][7]..[f]} evap{[1][2][3]..[8]}
      evar{[1][3][5][7]} [[!]seq]
```

[Parameters]

brk|seqclr|seq1|seq2|seq3|seq4|trcs1trcs2|trcr|trg|match:
Specifies a condition with which the event is to be combined.

brk:	Specifies a break condition.
seqclr:	Specifies a sequential clear condition.
seq1:	Specifies a first-step sequential condition.
seq2:	Specifies a second-step sequential condition.
seq3:	Specifies a third-step sequential condition.
seq4:	Specifies a fourth-step sequential condition.
trcs1:	Specifies a trace section on condition.
trcs2:	Specifies a trace section off condition.
trcr:	Specifies a trace qualify condition.
trg:	Specifies a trigger output condition.
match:	Specifies a trace trigger condition.
evep{[1][2][3]..[g]}:	Specifies the corresponding event specified by the eve command as a point by itself. Specifying this parameter with no numeric characters cancels the setting. (Nx85ET can use only 1-8 ch.). Each number corresponds to a channel number specified by eve(.[1][2][3]..[16])
ever{[1][3][5]..[f]}:	Specifies each pair of events specified by the eve command as an area. Specifying this parameter with no numeric characters cancels the setting. (Nx85ET can use only 1,3,5,7ch.)
1:	Specifies the conditions of channels 1 and 2 specified by eve as a range (and condition).
3:	Specifies the conditions of channels 3 and 4 specified by eve as a range (and condition).
5:	Specifies the conditions of channels 5 and 6 specified by eve as a range (and condition).
7:	Specifies the conditions of channels 7 and 8 specified by eve as a range (and condition).
9:	Specifies the conditions of channels 9 and 10 specified by eve as a range (and condition).
b:	Specifies the conditions of channels 11 and 12 specified by eve as a range (and condition).
d:	Specifies the conditions of channels 13 and 14 specified by eve as a range (and condition).
f:	Specifies the conditions of channels 15 and 16 specified by eve as a range (and condition).
evap{[1][2][3]..[8]}:	Specifies the corresponding event specified by the eva command as a point by itself. Specifying this parameter with no numeric characters cancels the setting. Each number corresponds to a channel number specified by eva([1][2][3]..[8]). (Nx85ET can use only 1-4 ch.)
evar{[1][3][5][7]}:	Specifies each pair of events specified by the eva command as an area. Specifying this parameter with no numeric characters cancels the setting. (Nx85ET can use only 1,3 ch.)

1:	Specifies the conditions of channels 1 and 2 specified by eva as a range (and condition).
3:	Specifies the conditions of channels 3 and 4 specified by eva as a range (and condition).
5:	Specifies the conditions of channels 5 and 6 specified by eva as a range (and condition).
7:	Specifies the conditions of channels 7 and 8 specified by eva as a range (and condition).

[!]seq:	Specifies a sequential condition.
seq:	Specifies a sequential condition. Enter ! to cancel the sequential condition. ! cannot be specified for a seq-related condition (seqclr, seq1, seq2, seq3, or seq4).

[Function]

The evt command specifies the use of each event specified by eve or eva.

[Examples]

evt brk evep1234 ever5 evap12 evar3

As break events, the events specified for channels 1 to 4 by eve are used as points; those specified for channels 5 and 6 as a range condition; those specified for channels 1 and 2 by eva as points; and those specified for channels 3 and 4 as a range.

evt brk evep ever evap evar

The events specified for evep, ever, evap, and evar as break events are canceled.

[Remark]

For the details of the sequential conditions, see the description of the seq command.

For the details of the trace section and qualify conditions, see Capture 8 "Details of Trace Functions".

extbrk command

[Format]

extbrk [disable|posi|nega]

[Parameters]

disable:	Disables this capability (default).
posi:	Break request at positive edge detection
nega:	Break request at negative edge detection

[Function]

The extbrk command specifies the break request using eternal input signal (1 pin of EXT connector (RSV-IN0)).

[Examples]

extbrk posi

A break is requested at positive edge detection.

Remark This command is not available for RTE-100-TP.
To use this capability, A12 pin of JTAG/N-Wire connector needs to be connected to CPU DBINT port.

help command

[Format]

```
help  [command]
```

[Parameters]

command:	Specifies the name of the command for which you require help. If this parameter is omitted, a list of commands is displayed.

[Function]

The help command displays a help message for a specified command.

[Examples]

```
help map
```

A help message for the map command is displayed.

inb, inh, and inw commands

[Format]

inb [ADDR]

inh [ADDR]

inw [ADDR]

[Parameters]

ADDR: Specifies the address of an input port in hexadecimal.

[Function]

The inb, inh, and inw commands read the I/O space in different sizes.

The inb command accesses I/O space in bytes, inh in half words, and inw in words.

[Examples]

inb 1000

The I/O space is read in bytes (8-bit units), starting at 1000H.

inh 1000

The I/O space is read in half words (16-bit units), starting at 1000H.

inw 1000

The I/O space is read in words (32-bit units), starting at 1000H.

init command

[Format]

init

[Parameters]

None

[Function]

The init command initializes KIT-NB85E-TP. All environment values are initialized.
A memory cache rejection area is not initialized.

jread command

[Format]

```
jread [ADDR [LENGTH]]
```

[Parameters]

ADDR:	Specifies an address in hexadecimal.
LENGTH:	Specifies the number of bytes to be read, in hexadecimal. (Max: 100h)

[Function]

The jread command reads the ROM emulation area allocated by the ROM command, via JTAG (the CPU). (Access to the ROM emulation area by ordinary commands is performed directly on internal memory.)

[Examples]

```
jread 100000 100
```

100h bytes, starting at 100000h, are read via JTAG.

nc command

[Format]

nc [[ADDR [LENGTH]]

[Parameters]

ADDR:	Specifies the start address of a memory cache rejection area.
LENGTH:	Specifies the length of the memory cache rejection area in bytes. The default value is 32 bytes. The allowable minimum value is also 32 bytes.

[Function]

To ensure quick memory access, KIT-NB85E-TP provides a memory read cache of 8 blocks*32 bytes. When the same memory address is accessed more than once, the read operation is not actually performed. This cache operation conflicts with the actual operation when an I/O unit is mapped onto memory. In such a case, specify a memory cache rejection area by using the nc command. Up to eight blocks can be specified as a memory cache rejection area. The allowable minimum block size is 32 bytes. Addresses ffff000h through fffffffh and 3fff000h through 3ffffffh constitute sfr areas of the internal ROM. As the default value, these areas are excluded.

[Examples]

nc 10000 100

A 100-byte area, starting at 10000h, is specified as a memory cache rejection area.

```
>nc 100000 100
 No Memory Cache Area
 No.  Address     Length
 1    00100000    00000100
 2    0ffff000    00001000
 3    03fff000    00001000
```

ncd command

[Format]

ncd block-number

[Parameters]

block-number: Specifies the block number for a memory cache rejection area to be deleted.

[Function]

The ncd command deletes a memory cache rejection area. Specify the block number corresponding to the memory cache rejection area to be deleted.

[Examples]

ncd 1

Block 1 is deleted from the memory cache rejection area.

```
>nc  100000   100
 No Memory Cache Area
 No.   Address      Length
 1     00100000     00000100
 2     0ffff000     00001000
 3     03fff000     00001000

>ncd 1
 No Memory Cache Area
 No.   Address      Length
 1     0ffff000     00001000
 2     03fff000     00001000
```

nsbp command

[Format]

```
nsbp  [[ADDR [LENGTH]]
```

[Parameters]

ADDR:	Specifies the start address of a software break prohibition area.
LENGTH:	Specifies the length of a software break prohibition area in bytes. The minimum unit of a specification area is the boundary of half word. The number of the areas which can be specified is a maximum of four.

[Function]

The nsbp command specifies an area to forbid a software break.
When a break point is specified, a debugger implicitly performs a memory test (write access) to an object address.
The state of some flash ROM may change by performing write access and right data may not be read.
When this happens, please forbid a software break by this command to prohibit use of write cycles.
Usually, it is not necessary to specify.

[Examples]

nsbp 10000 20000

A 20000-byte area, starting at 10000h, is specified as a software break prohibition area.

```
>nsbp 100000 20000
Num  Address     Length
01   00100000    00020000
```

nsbpd command

[Format]

nsbpd [block-number|/all]

[Parameters]

block-number:	Specifies the block number of the software break prohibition area to be deleted.
/all:	Specifies all software break prohibition area to be deleted.

[Function]

The nsbpd command deletes the software break prohibition area specified by nsbp.

[Examples]

nsbpd 1

Block 1 is deleted from a software break prohibition area.

```
>nsbp
Num  Address    Length
01   00100000   00200000
02   00400000   00010000

>nsbpd 1
Num  Address    Length
01   00400000   00010000
```

nrom command

[Format]

```
nrom  [[ADDR [LENGTH]]
```

[Parameters]

ADDR:	Specifies the start address of a forced user area.
LENGTH:	Specifies the length of a forced user area in bytes.

The minimum unit of the a specification area is as follows.

RTE-1000-TP:	4 bytes	
RTE-2000-TP:	Depends on the size of the ROM being emulated.	
	8/16 bits:	128K bytes
	32 bits:	256K bytes
	(64 bits:	512K bytes)

The number of areas which can be specified is a maximum of four.

[Function]

The nrom command specifies the area when part of ROM emulation area specified by ROM command is mapped to other resources on a user system. Usually, it is not necessary to specify this command. The operations for the specified area are as follows.

- An access from the debugger is forcibly made to the user system.
- The EMEMEN- signal is deasserted inactive (high level) in the cycle for accessing this area during execution (RTE-2000-TP only).

[Examples]

nrom 0 20000

A 20000-byte area, starting at 0h, is specified as a forced user area.

```
>nrom 0 20000
No.   Address     Length
  1   00000000    00020000

>nrom 100000 40000
No.   Address     Length
  1   00000000    00020000
  2   00100000    00040000
```

nromd command

[Format]

nromd [block-number|/all]

[Parameters]

block-number:	Specifies the block number for the forced user area to be deleted.
/all:	Specifies all the forced user area to be deleted.

[Function]

The nromd command deletes the forced user area specified by nrom.

[Examples]

nromd 1

Block 1 is deleted from the forced user area.

```
>nrom 1000000 40000
No.   Address     Length
  1   00000000    00020000
  2   00100000    00040000

>nromd 1
No.   Address     Length
  1   00100000    00040000
```

outb, outh, and outw commands

[Format]

```
outb [[ADDR] DATA]
outh [[ADDR] DATA]
outw [[ADDR] DATA]
```

[Parameters]

ADDR:	Specifies the address of an output port in hexadecimal.
DATA:	Specifies the data to be output in hexadecimal.

[Function]

The outb, outh, and outw commands write data to the I/O space in different sizes.
The outb command accesses the I/O space in bytes, outh in half words, and outw in words.

[Examples]

```
outb 1000 12
```

Byte data 12h is written to 1000H in the I/O space.

```
outh 1000 1234
```

Half word data 1234h is written to 1000H in the I/O space.

```
outw 1000 12345678
```

Word data 12345678h is written to 1000H in the I/O space.

reset command

[Format]

reset

[Parameters]

None

[Function]

The reset command resets the emulation CPU of KIT-NB85E-TP.

rom command (for RTE-1000-TP)

[Format]

```
rom [ADDR [LENGTH]] [512k|1m|2m|4m|8m|16m|32m|64m|128m|256m] [rom8|rom16]
    [bus8|bus16|bus32]
```

[Parameters]

ADDR [LENGTH]:	Specifies an area to be emulated.
ADDR:	Specifies a start address. An error occurs if the specified start address does not match the lowest address of the ROM to be emulated (boundary of the ROM).
LENGTH:	Specifies the number of bytes of the ROM to be emulated. (Must be specified in boundary units of 4 bytes.)
512k|1m|2m|4m|8m|16m|32m|64m|128m|256m:	Specifies the bit size of the ROM to be emulated. Sizes from 512K bits to 256M bits can be specified. For the 27C1024, for example, specify 1M bit.
rom8|rom16:	Specifies the number of data bits of the ROM to be emulated. Either 8 bits or 16 bits can be specified. If a DIP-32-ROM cable is used, choose rom8; if a DIP-40/42-ROM or STD-16BIT-ROM cable is used, choose rom16.
bus8|bus16|bus32:	Specifies the ROM bus size in the system to be emulated. 8 bits, 16 bits, or 32 bits can be specified.

[Function]

The rom command sets the ROM emulation environment of RTE-1000-TP. ADDR and LENGTH must be input in pairs. Input other parameters only when their values need to be changed. Parameters may be entered in any order. If the same parameter is entered twice, only the last entry is valid. The initial value of LENGTH is 0 (not used).

[Examples]

```
rom 100000 40000 1m rom16 bus16
```

The 256K bytes (40000h) of the 27C1024 (16-bit ROM with a size of 1M bit), starting at 100000h are emulated. Consequently, two 16-bit ROMs are emulated.

```
rom 0 40000 2m rom16 bus32
```

The 256K bytes (40000h) of the 27C2048 (16-bit ROM with a size of 2M bits), starting at 0x0, are emulated. Consequently, two 16-bit ROM is emulated.

<Remark>

Note on area specified by rom command

Access to a range specified by the rom command from the debugger is a direct access to the emulation memory in the tool. As a result, display is performed correctly even if the processor cannot correctly access ROM. It is therefore recommended to read and check data by using the jread command (that reads data via the CPU bus) or write data by setting verify to ON with the env command (download) in the initial stage of debugging.

rom1..rom4 commands (for RTE-2000-TP)

[Format]

```
rom1  [ADDR [LENGTH]] [512k|1m|2m|4m|8m|16m|32m|64m|128m|256m] [rom8|rom16]
      [bus8|bus16|bus32|bus64] [[!]wren]
rom2  [ADDR [LENGTH]] [512k|1m|2m|4m|8m|16m|32m|64m|128m|256m] [rom8|rom16]
      [bus8|bus16] [[!]wren]
rom3  [ADDR [LENGTH]] [512k|1m|2m|4m|8m|16m|32m|64m|128m|256m] [rom8|rom16]
      [bus8|bus16|bus32] [[!]wren]
rom4  [ADDR [LENGTH]] [512k|1m|2m|4m|8m|16m|32m|64m|128m|256m] [rom8|rom16]
      [bus8|bus16] [[!]wren]
```

rom1: This command performs setting of a module including the EMEM board mounted to slot #3.

rom2: This command performs setting of a module including the EMEM board mounted to slot #4.

rom3: This command performs setting of a module including the EMEM board mounted to slot #5.

rom4: This command performs setting of a module including the EMEM board mounted to slot #6.

[Parameters]

ADDR [LENGTH]:	Specifies an area to be emulated.
ADDR:	Specifies a start address. An error occurs if the specified start address does not match the lowest address of the ROM to be emulated (boundary of the ROM).
LENGTH:	Specifies the number of bytes of the ROM to be emulated. (Must be specified in boundary units of 4 bytes.)
512k|1m|2m|4m|8m|16m|32m|64m|128m|256m:	Specifies the bit size of the ROM to be emulated. Sizes from 512K bits to 256M bits can be specified. For the 27C1024, for example, specify 1M bit.
rom8|rom16:	Specifies the number of data bits of the ROM to be emulated. Either 8 bits or 16 bits can be specified. If a DIP-32-ROM cable is used, choose rom8; if a DIP-40/42-ROM or STD-16BIT-ROM cable is used, choose rom16.
bus8|bus16|bus32|bus64:	Specifies the ROM bus size in the system to be emulated. 8 bits, 16 bits, 32 bits, or 64 bits can be specified. >> [bus64] is a parameter for future use. (It is not used with KIT-NB85E-TP.)
[[!]wren]:	Write Enable: This setting is for using the emulation memory as RAM. wren enables writing, and !wren disables writing. The default value is !wren.

[Function]

The rom1 to rom4 commands set the ROM emulation environment of RTE-2000-TP. ADDR and LENGTH must be input in pairs. Input other parameters only when their values need to be changed. Parameters may be entered in any order. If the same parameter is entered twice, only the last entry is valid. The initial value of LENGTH is 0 (not used).

[Examples]

rom1 100000 40000 2m rom16 bus16 !wren

Slot position of EMEM board	Address range	Bus width	ROM		Write enable
			Bus width	Bits	
#3	100000 - 13ffff	16 bits	16 bits	2M bits	Disabled

rom2 140000 40000 2m rom16 bus16 wren

Slot position of EMEM board	Address range	Bus width	ROM		Write enable
			Bus width	Bits	
#4	140000 - 17ffff	16 bits	16 bits	2M bits	Enabled

rom1 0 80000 2m rom16 bus32 !wren

Slot position of EMEM board	Address range	Bus width	ROM		Write enable
			Bus width	Bits	
#3 + #4	000000 - 07ffff	32 bits	16 bits	2M bits	Disabled

Do not issue the rom2 command at this time.

<Remark>

Note on area specified by rom command

Access to the range specified by the rom1..rom4 commands from the debugger is a direct access to the emulation memory in the tool. As a result, display is performed correctly even if the processor cannot correctly access ROM. It is therefore recommended to read and check data by using the jread command (that reads data via the CPU bus) or write data by setting verify to ON with the env command (download) in the initial stage of debugging.

Relationship between rom command and EMEM board

rom command	Bus width	Slot position of EMEM board	Unusable rom command
rom1	8 bits	#3	
	16 bits	#3	
	32 bits	#3 + #4	rom2
	64 bits	#3 + #4 + #5 + #6	rom2, rom3, rom4
rom2	8 bits	#4	
	16 bits	#4	
rom3	8 bits	#5	
	16 bits	#5	
	32 bits	#5 + #6	rom4
rom4	8 bits	#6	
	16 bits	#6	

seq command

[Format]

seq [PASS] [step{1|2|3|4}]

[Parameters]

PASS:	Specifies in decimal the number of times the sequence condition is to be satisfied.
step{1|2|3|4}:	Specifies the number of steps in the sequence.
step1:	seq4->pass_count_decrement
step2:	seq3->seq4->pass_count_decrement
step3:	seq2->seq3->seq4->pass_count_decrement
step4:	seq1->seq2->seq3->seq4->pass_count_decrement

[Function]

The seq command sets the sequential conditions.
Use eve, eva, and evt to specify conditions for seq1 to seq4.
When the seqclr condition is satisfied during a sequence, the sequence is executed from the beginning.

[Example]

seq 100 step1

A seq event occurs when conditions seq1 -> seq2 -> seq3 -> seq4 are satisfied 100 times.

sfr command

[Format]

sfr [reg [VAL]]

[Parameters]

VAL: Specifies the value for an SFR register in hexadecimal.

reg: Specifies an SFR register name.
The following names can be used as register names:

Read/write registers:

CSC0 CSC1 BPC BSC BEC BHC VSWC
DSA0L DSA0H DDA0L DDA0H DSA1L DSA1H DDA1L DDA1H
DSA2L DSA2H DDA2L DDA2H DSA3L DSA3H DDA3L DDA3H
DBC0 DBC1 DBC2 DBC3
DADC0 DADC1 DADC2 DADC3 DCHC0 DCHC1 DCHC2 DCHC3
DRST IMR0 IMR1 IMR2 IMR3
PIC0..PIC63
PSC BCT0 BCT1 DWC0 DWC1 BCC ASC PRC RWC
DRC0 SCR0 RFC0 RFS0 DRC1 SCR1 RFC1 RFS1
DRC2 SCR2 RFC2 RFS2 DRC3 SCR3 RFC3 RFS3
DRC4 SCR4 RFC4 RFS4 DRC5 SCR5 RFC5 RFS5
DRC6 SCR6 RFC6 RFS6 DRC7 SCR7 RFC7 RFS7
ICC ICI ICD

Write-only registers:

PRCMD

Read-only registers:

DDIS ISPR

[Function]

The sfr command sets and displays the value of the SFR register.

[Examples]

sfr PIC0

The value of the PIC0 register is displayed.

sfr PIC0 2

The value 2h is set in the PIC0 register.

symfile and sym commands

[Format]

symfile FILENAME

sym [NAME]

[Parameters]

FILENAME:	Specifies file name.
NAME:	Specifies first character string in the symbols to be displayed.

[Function]

The symfile command reads symbols from the elf file specified by the FILENAME parameter.
Only global symbols can be read.
The sym command displays up to 30 symbols that have been read.

[Examples]

symfile c:\test\dry\dry.elf

Symbols are read from the elf file dry.elf in the c:\test\dry directory.

sym m

Up to 30 symbols that begin with "m" are displayed.

tp command

[Format]

tp [ADDR]

[Parameters]

ADDR: Specifies an even-numbered address in hexadecimal. (A0 is always corrected to 0.)

[Function]

The tp command specifies a trace trigger point.

Trace is used to monitor the execution status before and after a trigger point. (For information on how to use the trigger point, refer to the description of the tron command.)

[Examples]

tp 100000

The execution of the instruction at 100000h is specified as a trigger point.

[Note]

If delay mode is specified with the tron command, the trigger point specification is ignored.

Delay mode can be canceled by entering tron !delay.

tsp1 and tsp2 commands

[Format]

tsp{1|2} [ADDR] [asid ASID|noasid] [/del]

[Parameters]

tsp{1|2}:	Input before the condition of tsp1 or tsp2 is specified.
ADDR:	Specifies an execution address in hexadecimal.
asid ASID|noasid:	For future expansion. Use noasid.
/del:	Clears the specified address.

[Function]

The tsp1 and tsp2 commands specify the switch points (addresses) of the two trace points.
The condition in which the trace information is to be loaded can be changed by using the specified switch point. (For information on how to specify the loading condition, refer to the description of the tron command.)

[Examples]

tsp1 100000

The execution of the instruction at 100000h is specified as a switch point.

[Remark]

The switch point specified by this command becomes valid when the tron command has been issued.

td1 and td2 commands

[Format]

```
td{1|2} [ADDR [MASK]] [asid ASID|noasid] [/del]
```

[Parameters]

td{1|2}:	Input before the condition of td1 or td2 is specified.
ADDR:	Specifies an address.
MASK:	Specifies the mask data of an address in hexadecimal. Bits that are 1 are not subject to comparison. Only bits 9 through 2 are valid.
asid ASID|noasid:	For future expansion. Use noasid.
/del:	Clears the specified address.

[Function]

The td1 and td2 commands set the conditions of the data access cycles to be recorded by trace. Trace loads execution history information and the access cycle of the address specified here.

[Examples]

```
td1 100000 ff
```

The access cycle of address 1000xxh is loaded to trace.

tron command

[Format]

```
tron  [DELAY] [[!]delay] [[!]real] [[!]force] {[[!]evttrcs1] [[!]evttrcs2] |
      [[!]evttrcr]} [tr1_{[0]..[h]}|tr1_all] [tr2_{[0]..[h]}|tr2_all]
      [[!]clock2] [[!]stop] [noext|posi|nega] [[!]td1] [[!]td2] [[!]debug]
```

[Parameters]

DELAY = 0..3fffd delay counter <Caution> (0..1fffd for RTE-1000-TP)
Specifies the number of frames in memory that are to be loaded in response to a trigger, in hexadecimal.

[!]delay: Specifies forced delay mode. Enter !delay to return to normal mode.
In forced delay mode, trace is forcibly terminated when the number of frames specified by the delay counter are traced after trace starts. In this mode, trigger events are ignored.

[!]real: Specifies the execution mode during trace. real specifies the real-time execution mode. The trace information may overflow in real-time execution mode. ! specifies the non-real-time execution mode. An overflow does not occur in this mode, but the execution speed drops.

[!]force: Specifies forced start of trace. If forced start is cleared by specifying !, the condition of tsp1 is assumed.

[[!]evttrcs1][[!]evttrcs2]|[[!]evttrcr]: Use the initial value (!) of this parameter.

tr1_{[0]..[h]}|tr1_all: Specifies the trace information to be loaded after the switch point of tsp1.
Usually, it is used in order to start taking in of trace by specifying tr1_all.

tr1_{[0]..[h]: 0: Interrupt, 1: Exception, 2: RETI, 3: JMP, 4: JR, 5: JARL,
6: Condition Jump (not taken), 7: Condition Jump (taken),
8: CALLT, 9: SWITCH, a: DISPOSE, b: CTRET,
c: td1 read cycle, d: td1 write cycle,
e: td2 read cycle, f: td2 write cycle,
g: tp, h: evt_match

tr1_all: Loads all trace information.

tr2_{[0]..[h]}|tr2_all: Specifies the trace information to be loaded after the switch point of tsp2.
Usually, it is used in order to stop taking in of trace by specifying nothing.

tr2_{[0]..[h]: 0: Interrupt, 1: Exception, 2: RETI, 3: JMP, 4: JR, 5: JARL,
6: Condition Jump (not taken), 7: Condition Jump (taken),
8: CALLT, 9: SWITCH, a: DISPOSE, b: CTRET,
c: td1 read cycle, d: td1 write cycle,
e: td2 read cycle, f: td2 write cycle,
g: tp, h: evt_match

tr2_all: Loads all trace information.

[!]clock2: Specifies the trace sampling clock. clock2 specifies 1/2 of VBCLK. ! specifies 1/1.
Usually, use !clock2.

[!]stop: Specifies trace output in the stop mode. stop stops trace in the stop mode.
! does not stop trace.

noext|posi|nega: Specifies an external input pin (EXI0) as a trigger.

noext: Does not use EXI0 as a trigger.

posi: Uses the rising edge of EXI0 as a trigger.

nega: Uses the falling edge of EXI0 as a trigger.

[!]td1: Specifies Trace Data Condition 1 (td1) as trigger. ! clears the setting.

[!]td2: Specifies Trace Data Condition 2 (td2) as trigger. ! clears the setting.

Remark [[!]td1][[!]td2] is not available for RTE-100-TP.
If the condition of td1 and td2 are overlapped during that cycle, specify td1 as trigger condition. If td2 is specified in such case, the trigger might not work correctly.

[!]debug: Always use the initial value (!debug) of this parameter.

[Function]

The tron command clears the trace buffer and the settings of trace, and begins loading trace data.

[Examples]

Unconditionally traces <u>3fffd</u> cycles immediately after tron in the <u>delay</u> mode.

```
>tron delay 1ffff                                << Start of trace
 Trace Settings:
 Delay Count      = 0003fffd
 Trace Mode       = Real Time (real)
 Start Mode       = Force Start (force)
 Delay Mode       = Enable (delay)
 Event trcs1      = Disable (!evttrcs1)
 Event trcs2      = Disable (!evttrcs2)
 Event trcr       = ------
 Sampling cond1   = tr1_0123456789abcdefgh
 Sampling cond2   = tr2_0123456789abcdefgh
 Trace Clock      = VBCLK (!clock2)
 STOP Mode        = Disable (!stop)
 Ext Trigger      = Disable (noext)
 TD1 Trigger      = Disable (!td1)
 TD2 Trigger      = Disable (!td2)
 Debug Mode       = Disable (!debug)
```

Traces loading after trigger in 1ffff cycles by using execution of the instruction at address <u>100000h</u> as a trigger.

```
>tp 100000                                       <<Trigger specification
 Trigger Point Settings:
    Address   AISD
 tp 00100000 noasid

>tron !delay 1ffff                               <<Start of trace
 Trace Settings:
 Delay Count      = 0001ffff
 Trace Mode       = Real Time (real)
 Start Mode       = Force Start (force)
 Delay Mode       = Disable (!delay)
 Event trcs1      = Disable (!evttrcs1)
 Event trcs2      = Disable (!evttrcs2)
 Event trcr       = ------
 Sampling cond1   = tr1_0123456789abcdefgh
 Sampling cond2   = tr2_0123456789abcdefgh
 Trace Clock      = VBCLK (!clock2)
 STOP Mode        = Disable (!stop)
 Ext Trigger      = Disable (noext)
 TD1 Trigger      = Disable (!td1)
 TD2 Trigger      = Disable (!td2)
 Debug Mode       = Disable (!debug)
```

Traces the execution history from execution of address <u>100000h</u> to execution of address <u>100100h</u>, using tsp1 as the trace start condition and tsp2 as the trace stop condition.

```
>tsp1 100000                          << Sets point to be used as a start condition.
 Trace Switch Point Settings:
       Address   AISD
 tsp1 00100000 noasid
 tsp2 /del

>tsp2 100100                          << Sets point to be used as a stop condition.
 Trace Switch Point Settings:
       Address   AISD
 tsp1 00100000 noasid
 tsp2 00100100 noasid

>tron !force tr1_al| tr2_             << Specifies all for tsp1 and none for tsp2.
 Trace Settings:
 Delay Count       = 0000ffff
 Trace Mode        = Real Time (real)
 Start Mode        = Start by tsp1 or evttrcs1 or evttrcr (!force)
 Delay Mode        = Disable (!delay)
 Event trcs1       = Disable (!evttrcs1)
 Event trcs2       = Disable (!evttrcs2)
 Event trcr        = ------
 Sampling cond1    = tr1_0123456789abcdefgh
 Sampling cond2    = tr2_
 Trace Clock       = VBCLK (!clock2)
 STOP Mode         = Disable (!stop)
 Ext Trigger       = Disable (noext)
 TD1 Trigger       = Disable (!td1)
 TD2 Trigger       = Disable (!td2)
 Debug Mode        = Disable (!debug)
```

troff command

[Format]

```
troff
```

[Parameters]

None

[Function]

The troff command forcibly terminates the loading of trace data.

trace command

[Format]

trace [POS] [all|pc|data] [asm] [asm|ttag1|ttag2] [subNN]

[Parameters]

POS=±0..3fffd	Specifies the trace display start position in hexadecimal, assuming the vicinity of a trigger cycle or the ending cycle to be 0. <Caution> (0..1fffd for RTE-1000-TP)
all|pc|data	Specifies the cycle in loaded trace information that is to be displayed.
all:	All cycles
pc:	Execution cycles only
data:	Data cycles only
asm|ttag1|ttag2	Specifies the display type.
asm:	Displays assembled listing.
ttag1:	Displays assembled listing and Time Tag in absolute time format.
ttag2:	Displays assembled listing and Time Tag in relative time format.

Remark The ttag1|ttag2 specification is not available for RTE-100-TP.

subNN:	Number of instructions to be disassembled in succession from an information item to actually be loaded (hexadecimal). The initial value is 80h (sub80).

[Function]

The trace command displays the contents of the trace buffer.
Issuing this command during trace terminates the loading process.

[Display]

```
>trace asm -15
  Cycle     Sub   Address        Code      Instruction                 EXT   Stat
 -00001e    ----  00:0010558e  ffbfb7da  jarl     00100d68h          1111  JMPS  JARL
 -000014    ----  00:00100d68  3f460000  st.b     r7,+00h[r6]        1111  JMPD  JARL
* 000000    ----  --:00100d6c  007f      jmp      [lp]               1111  MATCH
  000002    ----  00:00105592  664003d0  movehi   03d0h,zero,r12     1111  JMPD  JMP
  000002    0001  00:00105596  672ca4b2  ld.h     -05b4eh[r12],r12   1111  SUB
  000002    0002  00:0010559a  6ecc0010  andi     0010h,r12,r13      1111  SUB
  000002    0003  00:0010559e  69e0      cmp      zero,r13           1111  SUB
  00000c    ----  00:001055a0  1d92      be       001055d2h          1111  JMPS  BcondNT
  000016    ----  00:001055a2  16400380  movehi   0380h,zero,r2      1111  JMPD  BcondNT

>trace -15 ttag1
  Cycle     Sub   Address        Code      Instruction      EXT        Stat
 -00001e    ----  00:0010558e  ffbfb7da   jarl    00100d68h          1111   JMPS  JARL
                               time = 000,001,448,264.9uS
 -000014    ----  00:00100d68  3f460000  st.b     r7,+00h[r6]        1111  JMPD  JARL
                               time = 000,001,448,265.3uS
* 000000    ----  --:00100d6c  007f       jmp     [lp]               1111  MATCH
                               time = 000,001,448,265.7uS
  000002    ----  00:00105592  664003d0  movehi   03d0h,zero,r12     1111  JMPD  JMP
                               time = 000,001,448,267.4uS
  000002    0001  00:00105596  672ca4b2  ld.h     -05b4eh[r12],r12   1111  SUB
  000002    0002  00:0010559a  6ecc0010  andi     0010h,r12,r13      1111  SUB
  000002    0003  00:0010559e  69e0      cmp      zero,r13           1111  SUB
  00000c    ----  00:001055a0  1d92      be       001055d2h          1111  JMPS  BcondNT
```

```
                                         time = 000,001,448,268.5uS
     000016    ----    00:001055a2  16400380   movehi  0380h,zero,r2      1111  JMPD  BcondNT
                                         time = 000,001,448,268.9uS

  >trace -15 ttag2
     Cycle     Sub     Address      Code       Instruction                EXT   Stat
    -00001e    ----     00:0010558e ffbfb7da   jarl    00100d68h          1111  JMPS  JARL
                                         time = 000,000,000,002.6uS
    -000014    ----    00:00100d68  3f460000   st.b    r7,+00h[r6]        1111  JMPD  JARL
                                         time = 000,000,000,000.4uS
  *  000000    ----     --:00100d6c 007f       jmp     [lp]               1111  MATCH
                                         time = 000,000,000,000.2uS
     000002    ----    00:00105592  664003d0   movehi  03d0h,zero,r12     1111  JMPD  JMP
                                         time = 000,000,000,001.7uS
     000002    0001    00:00105596  672ca4b2   ld.h    -05b4eh[r12],r12   1111  SUB
     000002    0002    00:0010559a  6ecc0010   andi    0010h,r12,r13      1111  SUB
     000002    0003    00:0010559e  69e0       cmp     zero,r13           1111  SUB
     00000c    ----    00:001055a0  1d92       be      001055d2h          1111  JMPS  BcondNT
                                         time = 000,000,000,001.1uS
     000016    ----    00:001055a2  16400380   movehi  0380h,zero,r2      1111  JMPD  BcondNT
                                         time = 000,000,000,000.4uS
```

Cycle:	Relative positions in the trace buffer are displayed in hexadecimal. The vicinity of the trigger point or the trace end frame is assumed to be 0.
Sub:	Cycle numbers generated by analyzing branching and number-of-executed-instruction information.
Address:	Execution addresses or bus cycle addresses are displayed.
Code:	Instruction code or bus cycle data is displayed.
Instruction:	Instruction mnemonics or bus types are displayed.
EXT:	The states of external input pins EXI3 to EXI0 are displayed as bit strings.
Stat:	The types of trace packets on which display is based are displayed.

TRGSTART0	START packet is generated. Sub-switch is set to ON.
TRGSTART1	START packet is generated. Sub-switch is set to OFF.
MATCH	MATCH packet is generated.
OVF	Overflow occurs.
TRCEND	TRCEND packet is generated.
JMPD <>	JMPD packet is generated. (< > will be explained later.)
JMPDS <>	JMPDS packet is generated. (< > will be explained later.)
JMPS <>	JMPS packet is generated. (< > will be explained later.)
OPCODE	Op code access (execution) occurs.
DATAW1, 2	Memory write occurs (trace packet).
DATAR1, 2	Memory read occurs (trace packet).
SUB	Sub-cycle

"<>" above indicates the following character strings. It indicates an instruction or an event that has caused branch.

NMI/INT	By occurrence of interrupt
EXP/TRAP	By occurrence of exception
RETI	By corresponding instruction
JMP	By corresponding instruction
JR	By corresponding instruction
JARL	By corresponding instruction

BcondNT	By corresponding instruction
Bcond	By corresponding instruction
CALLT	By corresponding instruction
SWITCH	By corresponding instruction
DISPOSE	By corresponding instruction
CTRET	By corresponding instruction
FSTART	Forced start of trace

* mark:	Trigger point (may shift slightly).
time =	Displays Time Tag

Remark The Time Tag reflects a value when the CPU outputs branch information. The output of branch information has some delay from the time of actual execution, and the delay is not constant. Thus, the measurement value of the Time Tag has some error. Especially, please ignore the measurement result immediately after the execution, as it is unreliable.

ver command

[Format]
ver

[Parameters]
None

[Function]
The ver command displays the version of KIT-NB85E-TP.