APPENDIX A. KIT-V850E/ME2-TP(-H) INTERNAL COMMANDS

This appendix describes the KIT-V850E/ME2-TP(-H) 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 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[xxx][.[yyy]]{M|K}] [[!]nmi0] [[!]nmi1] [[!]nmi2]
[rtrcb{0|25|50|75}] [nrtrcb{12|25|37|50}] [[!]iiram_chk]

[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.

[!]nmi0: Specifies whether pins NMI are to be masked. Enter ! if they are not to be masked.

[!]nmi1: It is not used in this KIT. Please do not change from the state of !nmi1.

[!]nmi2: It is not used in this KIT. Please do not change from the state of !nmi2.

[jtag[xxx][.[yyy]]{M|K}]: The frequency of a JTAG clock is specified in the unit of MHz or KHz.
Being set up is rounded by the value not more than that nearest to a specification value although any value can be specified. Actual set value can be checked by display.

RTE-2000-TP : [25MHz,12.5MHz,5MHz,2MHz,1MHz,500KHz,250KHz,100KHz]

RTE-2000H-TP: [125MHz,100MHz,80MHz,60MHz,50MHz,40MHz,30MHz,25MHz,12.5MHz, 5MHz,2MHz,1MHz,500KHz,250KHz,100KHz,50KHz,25KHz,10KHz]

> **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.
> Initial value is automatically set as the highest frequency which made 25MHz the maximum and which works.
> When you set it as the value more than initial value, please perform into the tolerance of CPU.
> The behavior at the time of setting up the frequency more than the tolerance of CPU cannot be guaranteed.

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.

[!]iiram_chk: If it starts execution of a program from the address in instruction RAM, it is specification of whether to confirm that it is the read mode.
It is validity at iiram_chk. It is invalid at !iiram_chk.

[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(H)-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
IIRAM Settings:
 Mode Check   = Enable  (iiram_chk)
```

With RTE-2000(H)-TP

```
Probe:
 Unit            : RTE-2000(H)-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)
IIRAM Settings:
 Mode Check   = Enable  (iiram_chk)

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.

env JTAG40M

A JTAG clock is set as 40MHz.

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".

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 environment of ICE is initialized in the state at the time of starting.

All the configuration values are initialized except for the following.

- A memory cache rejection area (-> nc command)

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-V850E/ME2-TP(-H) provides a memory read cache of 8 blocks*32 bytes in the firmware. 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 ffffffh 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(H)-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(H)-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.

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(H)-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.
When the address in the middle of ROM is specified as a start address, the area below the specification address turns into a non-emulation area.

LENGTH: The byte count of ROM to emulate is specified.

Remark The minimum unit of the area which can be specified by ADDR and LENGTH is as follows according to the size of ROM currently emulated.
- 8 / 16-bit:128 k-byte unit
- 32-bit : 256 k-byte units
- 64-bit : 512 k-byte units

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.

[[!]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(H)-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:

SFR (R/W):
PAL PALL PAH PAHL PAHH PDH PDHL PDHH PCS PCT PCM PCD PMAL
PMALL PMAH PMAHL PMAHH PMDH PMDHL PMDHH PMCS PMCT PMCM PMCD PMCAL
PMCALL PMCAH PMCAHL PMCAHH PMCDH PMCDHL PMCDHH PMCCS PFCCS PMCCT PFCCT
PMCCM PFCCM PMCCD PFCDH PFCDHL PFCDHH PFCALL CSC0 CSC1 BEC BHC VSWC
ICC ICCL ICCH ICD 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 IMR0L
IMR0H IMR1 IMR1L IMR1H IMR2 IMR2L IMR2H IMR3 IMR3L IMR3H IMR4 IMR4L
IMR4H IMR5 IMR5L IMR5H P1IC0 P1IC1 P2IC1 P2IC2 P2IC3 P2IC4 P2IC5 P5IC0
P5IC1 P5IC2 P6IC5 P6IC6 P6IC7 PDIC0 PDIC1 PDIC2 PDIC3 PDIC4 PDIC5 PDIC6
PDIC7 PDIC8 PDIC9 PDIC10 PDIC11 PDIC12 PDIC13 PDIC14 PDIC15 PLIC0 PLIC1
OVCIC0 OVCIC1 OVCIC2 OVCIC3 OVCIC4 OVCIC5 CCC0IC0 CCC0IC1 CCC1IC0 CCC1IC1
CCC2IC0 CCC2IC1 CCC3IC0 CCC3IC1 CCC4IC0 CCC4IC1 CCC5IC0 CCC5IC1 CMDIC0
CMDIC1 CMDIC2 CMDIC3 CC10IC0 CC10IC1 CM10IC0 CM10IC1 OV1IC0 UD1IC0
CC11IC0 CC11IC1 CM11IC0 CM11IC1 OV1IC1 UD1IC1 DMAIC0 DMAIC1 DMAIC2
DMAIC3 CSI3IC0 COVF3IC0 CSI3IC1 COVF3IC1 UREIC0 URIC0 UTIC0 UIFIC0
UTOIC0 UREIC1 URIC1 UTIC1 UIFIC1 UTOIC1 ADIC US0BIC US1BIC US2BIC USP2IC
USP4IC RSUMIC PSC ADM0 ADM1 ADM2
ADTS P1 P2 P5 P6 P7 PM1 PM2 PM5
PM6 PM7 PMC1 PMC2 PMC5 PMC6 PMC7
PFC1 PFC2 PFC5 PFC6 PFC7 BCT0
BCT1 DWC0 DWC1 BCC ASC BCP LBS LBC0 LBC1 FWC FIC BMC PRC SCR1
RFS1 SCR3 RFS3 SCR4 RFS4 SCR6 RFS6 CMD0
TMCD0 CMD1 TMCD1 CMD2 TMCD2 CMD3
TMCD3 TMENC10 CM100 CM101 CC100
CC101 CCR10 TUM10 TMC10 SESA10 PRM10 NCW10 TMENC11
CM110 CM111 CC110 CC111 CCR11 TUM11 TMC11 SESA11 PRM11 NCW11
CCC00 CCC01 TMCC00 TMCC01 SESC0 NCWC0
CCC10 CCC11 TMCC10 TMCC11 SESC1 NCWC1
CCC20 CCC21 TMCC20 TMCC21 SESC2 NCWC2
CCC30 CCC31 TMCC30 TMCC31 SESC3 NCWC3
CCC40 CCC41 TMCC40 TMCC41 CCC50
CCC51 TMCC50 TMCC51 OSTS IRAMM
DTFR0 DTFR1 DTFR2 DTFR3 PSMR CKC CKS
UCKC SSCGC DTOC DIFC UB0CTL0
UB0CTL2 UB0STR UB0FIC0 UB0FIC1 UB0FIC2 UB0FIC2L UB0FIC2H UB1CTL0
UB1CTL2 UB1STR UB1FIC0 UB1FIC1 UB1FIC2 UB1FIC2L UB1FIC2H PWMC0
PWM0 PWML0 PWMH0 PWMC1 PWM1 PWML1 PWMH1 INTF1
INTF2 INTF5 INTF6 INTFAL INTFDH INTFDHL INTFDHH INTR1
INTR2 INTR5 INTR6 INTRAL INTRDH INTRDHL INTRDHH CSIM30
CSIC30 SFDB30 SFDB30L SFDB30H SFA30 CSIL30 SFN30 CSIM31
CSIC31 SFDB31 SFDB31L SFDB31H SFA31 CSIL31 SFN31 UF0CS
UF0BC UF0E0N UF0E0NA UF0EN UF0ENM UF0SDS UF0IM0
UF0IM1 UF0IM2 UF0IM3 UF0IM4 UF0IDR UF0DEND UF0GPR
UF0MODC UF0AIFN UF0AAS UF0E1IM UF0E2IM UF0E3IM UF0E4IM UF0E7IM UF0E8IM
UF0DSTL UF0E0SL UF0E1SL UF0E2SL
UF0E3SL UF0E4SL UF0E7SL UF0E8SL UF0ADRS UF0CNF UF0IF0 UF0IF1 UF0IF2
UF0IF3 UF0IF4 UF0DSCL UF0DD0 UF0DD1 UF0DD2 UF0DD3 UF0DD4 UF0DD5 UF0DD6
UF0DD7 UF0DD8 UF0DD9 UF0DD10 UF0DD11 UF0DD12 UF0DD13 UF0DD14 UF0DD15
UF0DD16 UF0DD17 UF0CIE0 UF0CIE1 UF0CIE2 UF0CIE3 UF0CIE4 UF0CIE5 UF0CIE6
UF0CIE7 UF0CIE8 UF0CIE9 UF0CIE10 UF0CIE11 UF0CIE12 UF0CIE13 UF0CIE14

```
  UF0CIE15 UF0CIE16 UF0CIE17 UF0CIE18 UF0CIE19 UF0CIE20 UF0CIE21 UF0CIE22
  UF0CIE23 UF0CIE24 UF0CIE25 UF0CIE26 UF0CIE27 UF0CIE28 UF0CIE29 UF0CIE30
  UF0CIE31 UF0CIE32 UF0CIE33 UF0CIE34 UF0CIE35 UF0CIE36 UF0CIE37 UF0CIE38
  UF0CIE39 UF0CIE40 UF0CIE41 UF0CIE42 UF0CIE43 UF0CIE44 UF0CIE45 UF0CIE46
  UF0CIE47 UF0CIE48 UF0CIE49 UF0CIE50 UF0CIE51 UF0CIE52 UF0CIE53 UF0CIE54
  UF0CIE55 UF0CIE56 UF0CIE57 UF0CIE58 UF0CIE59 UF0CIE60 UF0CIE61 UF0CIE62
  UF0CIE63 UF0CIE64 UF0CIE65 UF0CIE66 UF0CIE67 UF0CIE68 UF0CIE69 UF0CIE70
  UF0CIE71 UF0CIE72 UF0CIE73 UF0CIE74 UF0CIE75 UF0CIE76 UF0CIE77 UF0CIE78
  UF0CIE79 UF0CIE80 UF0CIE81 UF0CIE82 UF0CIE83 UF0CIE84 UF0CIE85 UF0CIE86
  UF0CIE87 UF0CIE88 UF0CIE89 UF0CIE90 UF0CIE91 UF0CIE92 UF0CIE93 UF0CIE94
  UF0CIE95 UF0CIE96 UF0CIE97 UF0CIE98 UF0CIE99 UF0CIE100 UF0CIE101 UF0CIE102
  UF0CIE103 UF0CIE104 UF0CIE105 UF0CIE106 UF0CIE107 UF0CIE108 UF0CIE109
  UF0CIE110 UF0CIE111 UF0CIE112 UF0CIE113 UF0CIE114 UF0CIE115 UF0CIE116
  UF0CIE117 UF0CIE118 UF0CIE119 UF0CIE120 UF0CIE121 UF0CIE122 UF0CIE123
  UF0CIE124 UF0CIE125 UF0CIE126 UF0CIE127 UF0CIE128 UF0CIE129 UF0CIE130
  UF0CIE131 UF0CIE132 UF0CIE133 UF0CIE134 UF0CIE135 UF0CIE136 UF0CIE137
  UF0CIE138 UF0CIE139 UF0CIE140 UF0CIE141 UF0CIE142 UF0CIE143 UF0CIE144
  UF0CIE145 UF0CIE146 UF0CIE147 UF0CIE148 UF0CIE149 UF0CIE150 UF0CIE151
  UF0CIE152 UF0CIE153 UF0CIE154 UF0CIE155 UF0CIE156 UF0CIE157 UF0CIE158
  UF0CIE159 UF0CIE160 UF0CIE161 UF0CIE162 UF0CIE163 UF0CIE164 UF0CIE165
  UF0CIE166 UF0CIE167 UF0CIE168 UF0CIE169 UF0CIE170 UF0CIE171 UF0CIE172
  UF0CIE173 UF0CIE174 UF0CIE175 UF0CIE176 UF0CIE177 UF0CIE178 UF0CIE179
  UF0CIE180 UF0CIE181 UF0CIE182 UF0CIE183 UF0CIE184 UF0CIE185 UF0CIE186
  UF0CIE187 UF0CIE188 UF0CIE189 UF0CIE190 UF0CIE191 UF0CIE192 UF0CIE193
  UF0CIE194 UF0CIE195 UF0CIE196 UF0CIE197 UF0CIE198 UF0CIE199 UF0CIE200
  UF0CIE201 UF0CIE202 UF0CIE203 UF0CIE204 UF0CIE205 UF0CIE206 UF0CIE207
  UF0CIE208 UF0CIE209 UF0CIE210 UF0CIE211 UF0CIE212 UF0CIE213 UF0CIE214
  UF0CIE215 UF0CIE216 UF0CIE217 UF0CIE218 UF0CIE219 UF0CIE220 UF0CIE221
  UF0CIE222 UF0CIE223 UF0CIE224 UF0CIE225 UF0CIE226 UF0CIE227 UF0CIE228
  UF0CIE229 UF0CIE230 UF0CIE231 UF0CIE232 UF0CIE233 UF0CIE234 UF0CIE235
  UF0CIE236 UF0CIE237 UF0CIE238 UF0CIE239 UF0CIE240 UF0CIE241 UF0CIE242
  UF0CIE243 UF0CIE244 UF0CIE245 UF0CIE246 UF0CIE247 UF0CIE248 UF0CIE249
  UF0CIE250 UF0CIE251 UF0CIE252 UF0CIE253 UF0CIE254 UF0CIE255
SFR (W):
  PRCMD UB0TX UB1TX UF0IC0 UF0IC1
```

[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..xxxx: delay counter
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 3fffd cycles immediately after tron in the delay 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 100000h 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 100000h to execution of address 100100h, 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..xxxx	Specifies the trace display start position in hexadecimal, assuming the vicinity of a trigger cycle or the ending cycle to be 0.
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.
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.

tdata_dly command

[Format]

tdata_dly [off|small|medium|large]

[Parameters]

off:	It is the correction value 0.
small:	It is the correction value 1.
medium:	It is the correction value 2. (initial value)
large:	It is the correction value 3.

[Function]

It is a command for adjusting the setup time of the trace data for a trace clock.

The setup time's off is the smallest and large becomes the largest.

A physical setup value should check the specification of the hardware of RTE-xxxx-TP to be used.

[Supplement]

Usually, although it is not necessary to change from initial value, depending on the state of CPU or a board, adjustment may be needed.

This command is a command which can be used only by RTE-2000(H)-TP.

ver command

[Format]

ver

[Parameters]

None

[Function]

The ver command displays the version of KIT-V850E/ME2-TP(-H).